SuperActivators: Only the Tail of the Distribution Contains Reliable Concept Signals

Cassandra Goldberg
University of Pennsylvania
cgoldber@seas.upenn.edu

Chaehyeon Kim
University of Pennsylvania
chaenyk@seas.upenn.edu

Adam Stein
University of Pennsylvania
steinad@seas.upenn.edu

Eric Wong
University of Pennsylvania
exwong@seas.upenn.edu

Abstract

Concept vectors aim to enhance model interpretability by linking internal representations with human-understandable semantics, but their utility is often limited by noisy and inconsistent activations. In this work, we uncover a clear pattern within the noise, which we term the **SuperActivator Mechanism**: while in-concept and out-of-concept activations overlap considerably, the token activations in the extreme high tail of the in-concept distribution provide a clear, reliable signal of concept presence. We demonstrate the generality of this mechanism by showing that SuperActivator tokens consistently outperform standard vector-based and prompting concept detection approaches—achieving up to a 14% higher F_1 score—across diverse image and text modalities, model architectures, model layers, and concept extraction techniques. Finally, we leverage these SuperActivator tokens to improve feature attributions for concepts. [1]

1 Introduction

Modern transformer-based models, while increasingly powerful and ubiquitous [1], remain opaque and can behave in ways that are unpredictable or harmful [2, 3]. This opacity hinders our ability to identify and debug undesirable representations—such as spurious correlations [4], biases [5], or fragile reasoning [6]—or to intervene when models produce undesirable outputs.

Concept vectors [7, 8], or semantically meaningful directions in a model's latent space, provide a lightweight tool for examining and influencing internal representations. They have been used to uncover hidden model failures [9, 10], and to steer model behavior away from hallucinations [11, 12], unsafe responses [13, 14], and toxic language [15, 16]. Unsupervised concept extraction is especially powerful, as it can reveal previously unknown knowledge embedded within model representations [17], while reducing reliance on costly labeled data.

To analyze the presence of concepts within a sample, we typically rely on their *activation scores*—a measure of alignment between an input token's embedding and a concept vector. However, these scores are often noisy and unreliable, and as a result misrepresent true concept presence. Prior works have shown that concept vectors frequently activate on unintended semantics [18, 19], generate overlapping signals for correlated concepts [20, 18], and exhibit unstable activation patterns across different model layers [21]. Figure 1 illustrates this ambiguity on an image of a dog reflected in a car mirror: the activation heatmaps for both the *Animal* and *Person* concepts appear to highlight the same

[1] Code released at https://github.com/BrachioLab/SuperActivators

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: Mechanistic Interpretability.

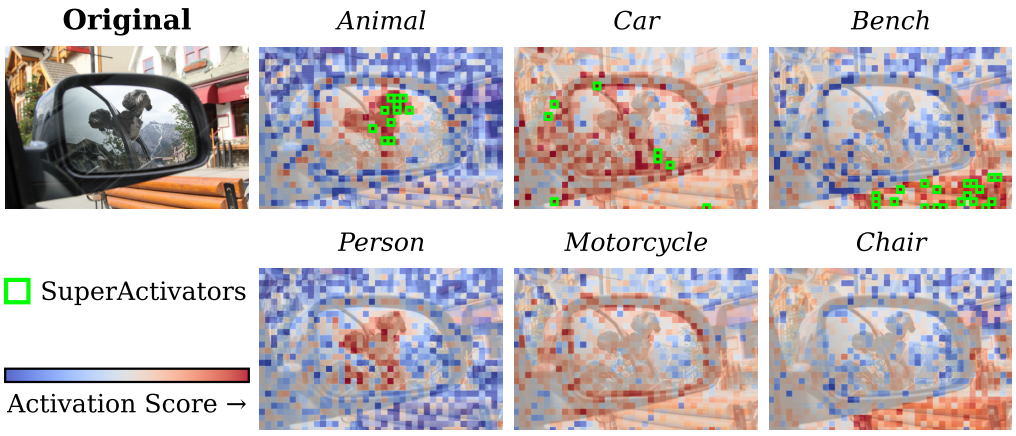


Figure 1: **The SuperActivator Mechanism concentrates the most informative concept signals into a sparse set of in-concept activations.** These signals reliably distinguish true concept occurrences even when concept activation heatmaps spuriously highlight absent concepts or fail to fully capture present ones. This example shows *LLaMA-3.2-11B-Vision-Instruct* linear separator concept activations on a COCO image; examples for all image and text datasets are provided in Appendix A.

region, even though only the former is present, and many tokens over the car region do not strongly activate for the *Car* concept. Such noisy activation signals makes it difficult to reliably detect or localize concepts.

To understand these inconsistencies more broadly, we analyze activation distributions for in-concept and out-of-concept tokens across multiple datasets. While the two distributions overlap considerably, we observe clear separation in the extreme high tail of the in-concept distribution. Notably, these high-activation tokens are well-distributed across in-concept samples, enabling them to reliably distinguish concept presence even when token activation maps are misleading or ambiguous. We term this behavior the **SuperActivator Mechanism** and show that it is a general property of how transformers encode semantics. Our analysis demonstrates that this mechanism more accurately detects concepts than standard concept-vector and prompting methods across various image and text modalities, model architectures, model layers, and concept extraction techniques. We also show that leveraging these localized signals leads to improved feature attributions for concepts.

Our key contributions are summarized as follows:

- **SuperActivator Mechanism:** By analyzing concept activation distributions across datasets, we discover that only the most highly activated tokens in the tail of the in-concept distribution are reliable indicators of concept presence. Using just a small set of these extreme activations, our method consistently outperforms standard vector- and prompt-based concept detection methods, improving F_1 scores by up to 14% absolute performance.
- **Broad Generality:** We show the SuperActivator Mechanism is a fundamental property of how transformers encode semantics, consistent across text and image modalities, model architectures, model layers, and both supervised and unsupervised concept extraction techniques.
- **Improved Concept Attributions:** Localizing concept signals with the SuperActivator Mechanism yields attribution maps with stronger alignment to ground-truth annotations and superior insertion/deletion performance relative to global concept-vector baselines.

2 Concept Vector Preliminaries

This section defines basic notation for representing inputs, concept vectors, and activation scores; more detailed formal definitions are provided in Appendix D.

Let f be a trained transformer model that processes an input sample $x \in \mathcal{X}$ (an image or a text sequence) through its layers. From any given layer of f, we can extract token-level embeddings $(z_1^{\text{tok}}(x), \ldots, z_{n(x)}^{\text{tok}}(x)) \in (\mathbb{R}^d)^{n(x)}$ and a sample-level embedding $z^{\text{cls}}(x) \in \mathbb{R}^d$. The number of

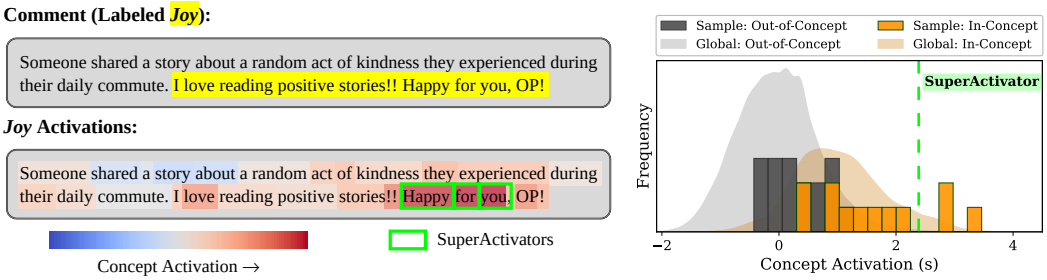


Figure 2: Transformers express concept activations inconsistently, making it difficult to distinguish in-concept tokens from out-of-concept tokens. In this test-set example from the *Augmented GoEmotions* dataset, the ground-truth span for *Joy* is highlighted, with token-level activations for LLaMA-Vision-Instruct-11B linear separator concepts shown both as a heatmap over the text (left) and as distributions (right). While a few in-concept tokens exhibit extremely high activations, many remain indistinguishable from out-of-concept token activations within the sample and across D_c^{out}.

tokens, $n(x)$, is sample dependent since it is influenced by text lengths and image sizes. For any semantic concept c, we associate a **concept vector** $v_c \in \mathbb{R}^d$, which represents a direction in the embedding space (see Section 4.1 for concept extraction methods). The concept activation score of an embedding z with respect to concept c is defined as the dot product of the embedding with the concept vector, $s_c(z) = \langle z, v_c \rangle$, where positive scores indicate alignment with the concept.

We are interested in characterizing, for each concept c, the distribution of activation scores across many samples. Let $\mathcal{D}_c^{\text{in}}$ and $\mathcal{D}_c^{\text{out}}$ denote the population-level distributions of activation scores for in-concept and out-of-concept tokens, respectively. Empirically, we estimate them using finite datasets D_c^{in} and D_c^{out} constructed from observed activations. Formally, let Z denote the set of all tokens across samples and $S_c = \{ s_c(z) : z \in Z \}$ their corresponding activation scores. If $Z_c^{\text{in}} \subseteq Z$ are the tokens labeled concept-positive for concept c and Z_c^{out} are the tokens drawn from samples that do *not* contain c (thus excluding out-of-concept tokens from samples containing c to avoid self-attention leakage), then

$$D_c^{\text{in}} = \{ s_c(z) : z \in Z_c^{\text{in}} \}, \qquad D_c^{\text{out}} = \{ s_c(z) : z \in Z_c^{\text{out}} \},$$

which serve as empirical samples from $\mathcal{D}_c^{\text{in}}$ and $\mathcal{D}_c^{\text{out}}$.

Concept activation scores are often leveraged for **concept detection** [22–24], which aims to determine whether a concept is present anywhere in a sample $x \in \mathcal{X}$. Because individual token activations vary across a sample, standard approaches apply an aggregation operator $G : \mathbb{R}^{n(x)+1} \to \mathbb{R}$ to obtain a per-sample concept activation score:

$$s_c^{\text{agg}}(x) = G\big(s_c(z_1^{\text{tok}}(x)), \ldots, s_c(z_{n(x)}^{\text{tok}}(x)), s_c(z^{\text{cls}}(x))\big).$$

The concept is considered detected if $s_c^{\text{agg}}(x)$ exceeds a threshold, typically obtained via calibration. There is no consensus on the best choice of aggregation operator G. Common strategies include using the score of the [CLS] token [16, 25], applying mean [26, 27] or max-pooling [28, 22], or using the score of the last token [29, 28].

Concept activations are also useful for **concept localization** (or attribution), which seeks to answer *where* a concept is located within a sample [30]. When evaluating concept localizations, we desire attribution maps that align with ground-truth annotations—segmentation masks for images or span-level labels for text. At the same time, attributions should be *faithful* [31], meaning that they accurately reflect the features that the model actually relies on.

3 The SuperActivator Mechanism Yields Clear Concept Signals Amid Noisy Concept Activations

3.1 Concept Activations are Inconsistent and Poorly Separated

Concept vectors promise interpretability but they often deliver noisy activations that are difficult to extract meaningful insights from. It is well-documented that concept vectors can encode spurious

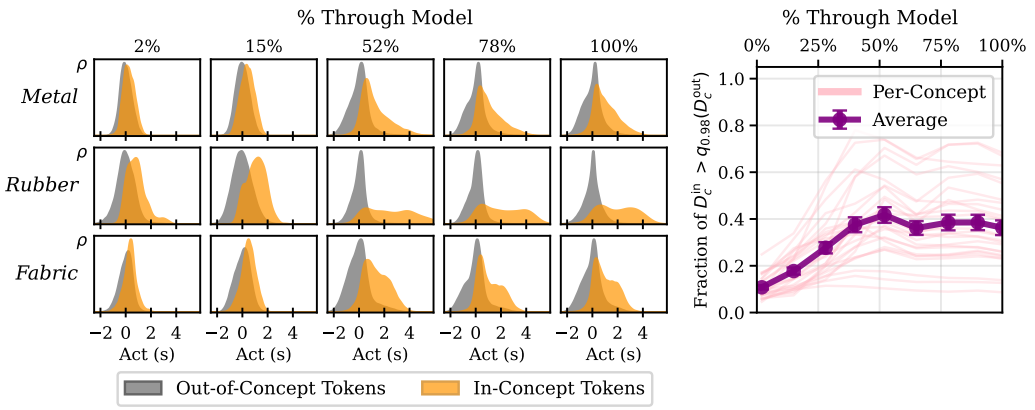


Figure 3: D_c^{in} and D_c^{out} become more distinct with depth, though the separation is concentrated in a small subset of tokens in the tail of D_c^{in}. Shown here are activation distributions for three linear separator concepts from *LLaMA-3.2-11B-Vision-Instruct* on the *OpenSurfaces* dataset (left), as well as the proportion of D_c^{in} activations exceeding $q_{0.98}(D_c^{\text{out}})$ across layers (right).

correlations and blur important context-specific distinctions [9, 32]. Other studies highlight issues of entanglement, where related features co-activate, and polysemanticity, where a single vector represents multiple unrelated concepts [20, 18, 19].

To study these limitations empirically, we focus our analysis on concept activations and their separability. In doing so, we identify a key challenge: many tokens labeled as concept-positive have activation scores that are not meaningfully different from those of concept-negative tokens. Figure 2 illustrates this problem: while a subset of in-concept tokens exhibit strong activations aligned with the concept *Joy*, a substantial portion fall well within the range of out-of-concept activations, both within the given text sample and across the broader distribution D_c^{out}. Consequently, no single threshold can partition the labeled *Joy* tokens from the other tokens.

We analyze activation behavior at the dataset level by characterizing the empirical activation distributions D_c^{in} and D_c^{out}. We also construct concepts at various model layers to examine how separability evolves throughout transformer models. The activations for linear separator concepts extracted from *LLaMA-3.2-11B-Vision-Instruct* on the *OpenSurfaces* dataset are visualized in Figure 3. At each layer, D_c^{out} appears roughly normal and centered at zero. In early layers, D_c^{in} overlaps considerably with D_c^{out}. As depth increases, the proportion of D_c^{in} exceeding the $98th$ percentile of D_c^{out}, $q_{0.98}(D_c^{\text{out}})$, grows steadily before plateauing in middle layers. This trend aligns with prior findings that concept representations become more separable at intermediate depths and can collapse again in the final layers due to task-specific compression [33, 25, 34].

The growing separation between D_c^{in} and D_c^{out} throughout the model does not result from a uniform shift of all in-concept activations. Instead, while many scores remain overlapping with the $98th$ percentile of D_c^{out}, and are thus largely indistinguishable from out-of-concept activations, D_c^{in} develops a heavy tail as a small subset of extreme activations become increasingly separable with depth.

Notably, we find that the high-activation tail of D_c^{in} exhibits good coverage: most true-concept samples contain at least one activation above the $98th$ percentile of D_c^{out}. This effect is shown for *LLaMA-3.2-11B-Vision-Instruct* linear separator concepts on the *OpenSurfaces* dataset in Figure 4, and we show that it generalized across datasets, models, and concept vector types in Appendix B.

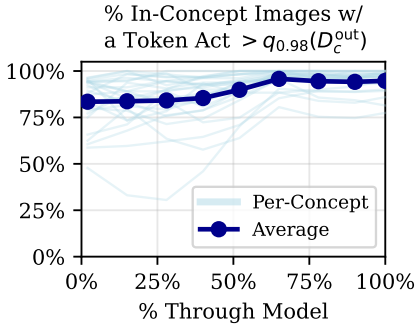


Figure 4: Most true-concept images in the *OpenSurfaces* dataset have at least one activation in the high-activation tail of D_c^{in}, well separated from $q_{0.98}(D_c^{\text{out}})$.

3.2 Introducing the SuperActivator Mechanism

A *reliable* concept signal should be *clear*, with activations that stand out from noise, and *accurate*, with high precision and broad coverage across true-concept samples. We find that such signals arise sparsely but consistently in the high-activation tail of D_c^{in}: they lie well outside D_c^{out} (Figure 3) and appear in most concept-positive samples (Figure 4). These results hold across modalities, architectures, and concept vector types, suggesting it is a general property of transformer representations.

We term this the **SuperActivator Mechanism**, where a small subset of extreme token activations carries the most reliable concept signals.

Formalizing SuperActivators: Let $\mathcal{S}^+_{\text{val},c} = \{\, s_c(z) : z \in Z_c^{\text{in}} \text{ from a validation set} \,\}$ be the empirical activation scores for concept c. For a sparsity level $\delta \in [0, 1]$, we define the *SuperActivator threshold* as

$$\tau^{\text{super}}_{c,\delta} \;=\; Q_{1-\delta}\big(\mathcal{S}^+_{\text{val},c}\big),$$

where $Q_q(S)$ denotes the q-quantile of a set of scores S. Tokens whose activations exceed this threshold form the set of SuperActivators,

$$\mathcal{T}^{\text{super}}_{c,\delta} \;=\; \{\, z \in Z_c^{\text{in}} : s_c(z) \geq \tau^{\text{super}}_{c,\delta} \,\}.$$

Intuitively, this means we are isolating the top δ percentage of the in-concept distribution D_c^{in}, i.e. tokens in its high-activation tail.

Leveraging SuperActivators for Concept Detection: We develop a SuperActivator-based aggregator that predicts the presence of c in a sample x if it contains at least one SuperActivator for that concept. Concretely, we apply a max-pooling operator $G_{\max}$ over token activations, predicting concept presence if $G_{\max}(s_c(z_1^{\text{tok}}(x)), \ldots, s_c(z_{n(x)}^{\text{tok}}(x))) \;\geq\; \tau^{\text{super}}_{c,\delta}$.

This approach is closely related to the standard max aggregator [22, 35], but instead of thresholding on the most activated token per sample, thresholds are derived from the globally most activated tokens across samples. This design enables direct control over sparsity, letting us study how detection performance varies with δ (See Appendix H and I). We find that SuperActivator detection is most effective at very low δ, showing that the most reliable concept information is concentrated in a small high-activation tail of D_c^{in}.

4 Concept Detection and Localization with SuperActivators

4.1 Experimental Setup

We evaluate our framework across different modalities, models and concept types.

Datasets. Vision datasets include CLEVR [36], COCO [37], and the PASCAL [38] and OPENSURFACES [39] sections of the BRODEN dataset [40]. For text, where token-level labels are scarce, we construct or augment three datasets: SARCASM, AUGMENTED iSARCASM [41], and AUGMENTED GOEMOTIONS [42]. Full details are provided in Appendix C.3.

Models. For images, we extract both patch and [CLS] token embeddings from the *CLIP* ViT-L/14 [43] and *LLaMA-3.2-11B-Vision-Instruct* [44]. For text, we use *LLaMA-3.2-11B-Vision-Instruct*, *Gemma-2-9B* [45], and *Qwen3-Embedding-4B* [46]. Since these models lack an explicit [CLS] token for text inputs, we approximate a [CLS]-style representation by averaging token embeddings, a strategy found to be effective in prior work [47–49].

Concept Types. We compute concepts at both the input token and [CLS]-level using the methods detailed in Appendix C.2: (1) mean prototypes [50], (2) labeled linear separators [7], (3) k-means [51, 34], (4) k-means-based separators, and (5) Sparse Autoencoders [19]. We incorporate the unsupervised concepts into our evaluation by matching each ground-truth concept with the discovered concept that is best at detecting it on a validation set. All methods in the following experiments make use of the same underlying concept vectors; detection strategies differ only in how activations are aggregated, while localization strategies generate attributions with respect to the same vectors.

Table 1: **Our SuperActivator-based method outperforms standard concept vector and prompting baselines on concept detection F_1 scores**. The results shown here are for linear separator concepts using the *LLaMA-3.2-11B-Vision-Instruct* model, where we improve performance by up to 14% over the best baseline. This trend generally holds across models and concept types, as detailed in Appendix E. **Bold** indicates the best score; underline marks the second best score.

	Concept Detection Methods					
	RandTok	LastTok [29]	MeanTok [26]	CLS [25]	Prompt [22]	SuperAct (Ours)
CLEVR	0.97 ± 0.09	0.88 ± 0.00	0.92 ± 0.00	0.96 ± 0.02	0.99 ± 0.01	**1.00 ± 0.00**
COCO	0.61 ± 0.01	0.68 ± 0.01	0.55 ± 0.01	0.57 ± 0.01	0.69 ± 0.05	**0.83 ± 0.01**
Surfaces	0.44 ± 0.01	0.41 ± 0.01	0.39 ± 0.01	0.46 ± 0.01	0.49 ± 0.06	**0.56 ± 0.02**
Pascal	0.66 ± 0.01	0.60 ± 0.01	0.59 ± 0.01	0.65 ± 0.01	0.68 ± 0.05	**0.82 ± 0.01**
Sarcasm	0.66 ± 0.06	0.68 ± 0.05	0.66 ± 0.06	0.74 ± 0.06	0.68 ± 0.07	**0.87 ± 0.04**
iSarcasm	0.89 ± 0.04	0.72 ± 0.03	0.79 ± 0.03	0.91 ± 0.03	0.79 ± 0.05	**0.92 ± 0.03**
GoEmot	0.37 ± 0.03	0.31 ± 0.03	0.19 ± 0.03	0.32 ± 0.03	0.25 ± 0.10	**0.46 ± 0.03**

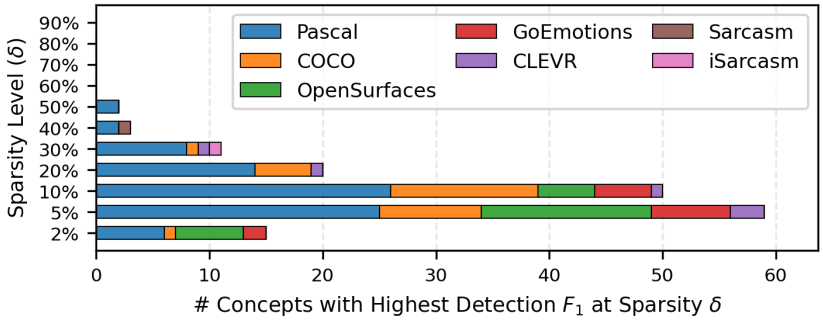


Figure 5: **SuperActivator-based concept detection is most effective when using only a small fraction of the most highly activated tokens (5–10%).** This figure presents the number of *LLaMA-3.2-11B-Vision-Instruct* linear separator concept vectors that achieve their strongest F_1 scores at each sparsity level δ. Comprehensive results are provided in Appendix H.

4.2 SuperActivators are Reliable Indicators of Concept Presence

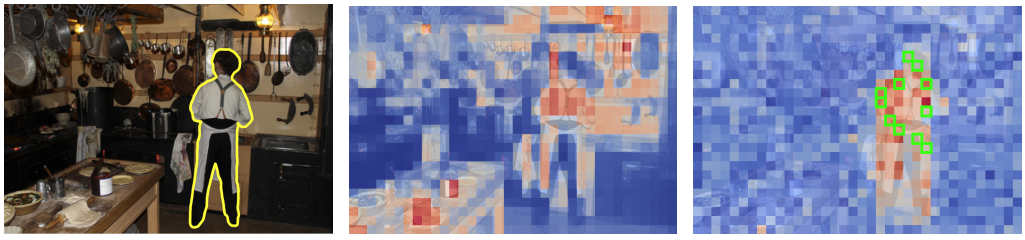
We now demonstrate that SuperActivator tokens serve as more reliable indicators of concept presence than both concept-vector baselines and prompting methods.

We compare against several baseline aggregation strategies: G_{CLS}, which selects the [CLS] activation [25]; G_{mean}, which averages input token activations [26]; G_{last}, which selects the final input token activation [29]; and G_{rand}, which selects a random token activation. We also include a prompting baseline, where *LLaMA-3.2-11B-Vision-Instruct* is directly queried about the presence of each concept, bypassing concept vectors altogether [22, 52, 28].

The sparsity δs and model layers used at test time are calibrated on a validation set to maximize each concept's detection F_1-score, following prior work showing that concept separability varies across layers [53–55]. Layer selection is performed independently for each detection method. To make this computationally feasible, calibration is performed over a fixed grid of layers (see Appendix C.1 for details). All reported detection scores reflect the average F_1 per dataset, obtained by averaging over concepts with weights proportional to their frequency in the test set.

As shown in Table 1, our SuperActivator method consistently outperforms all other detection strategies on LLaMA-3.2-11B-Vision-Instruct model linear separator concepts. Appendix E provides comprehensive results, demonstrating that this trend generally holds across all concept vector types and models that we experimented with. Prompting is typically the next strongest method, with [CLS]-token aggregators also showing competitive performance in certain settings.

Figure 5 summarizes the distribution of optimal sparsity levels δ across all *LLaMA-3.2-11B-Vision-Instruct* linear separator concepts. Performance typically peaks when using only a small fraction



(a) Original image (*Person* label in yellow)

(b) *Person* global concept vector attribution map

(c) *Person* local SuperActivators attribution map

Figure 6: **SuperActivators yield attribution masks that better align with the ground-truth concept region.** Shown are attribution maps for the concept *Person* in a COCO image using *LLaMA*-based linear separator concepts. Maps are computed with LIME attribution relative to (b) global concept vectors and (c) local SuperActivators, where red denotes high alignment and blue denotes low alignment. In (b), many high-attribution tokens lie outside the labeled *Person* region, while tokens inside the region often receive weak attribution. In contrast, (c) shows that attributions relative to the average embedding of local SuperActivators (highlighted by green boxes) correspond far more closely to the true *Person* area.

of the most activated tokens—2–10% for COCO, OPENSURFACES, and GOEMOTIONS, whereas ISARCASM peaks at a moderately higher 40%. These results indicate that only a sparse subset of tokens carry the strongest and most reliable concept information; including additional, weaker activations introduces noise from tokens that are less separated from D_c^{out}, diluting rather than improving performance. We note one nuance with Sparse Autoencoder concepts, where peak performance occurs at higher δ percentages, likely because SAEs already enforce sparsity during training. Detailed SAE-specific results and discussion are provided in Appendix L.1.

We perform several ablations to analyze how SuperActivator-based detection behaves across layers and sparsity levels. Appendix F shows heatmaps of average detection F_1 (weighted across concepts) for each model and dataset as a function of model depth, providing a global view of where concept signals are strongest. Appendix G summarizes the distribution of best-performing layers across concepts, revealing how different concepts peak at varying depths. To study sparsity, Appendix H reports histograms of optimal sparsity levels δ across model layers, while Appendix I plots F_1 as a function of δ at each concept's best-performing layer, showing how average SuperActivator detection performance varies with sparsity. Moreover, Appendix J provides qualitative examples of concept activations and SuperActivators (similar to Figure 1) across layers and datasets, illustrating how the SuperActivator mechanism manifests and evolves throughout transformer models.

Across image and text datasets, model architectures, and concept vector types, the same pattern emerges: the most reliable concept signals reside in the sparse, high-activation tail of D_c^{in}. The SuperActivator Mechanism thereby reflects a core principle of how transformers represent semantics.

4.3 SuperActivators Improve Attributions for Concepts

Standard concept attribution methods typically evaluate relevance with respect to a single global concept vector aggregated over many samples. While this captures broad concept information, it often blurs local context and introduces spurious correlations. In contrast, SuperActivators provide more consistent concept signals for detection (see Section 4.2), are tied to the specific local context of each sample, and avoid averaging across disparate occurrences. We hypothesize that using SuperActivators as the attribution objective improves attribution across three metrics: accuracy measuring average F_1 against ground truth, and insertion and deletion score based on the faithfulness metric.

To test this, we compare two attribution objectives: (1) the standard global concept vector and (2) our proposed method, which averages the embeddings of local SuperActivators within each instance.

We generate attribution maps following the standard procedures described in Appendix K.1, where attribution scores estimate each token's effect on changes in a given objective. Conventional concept attribution methods use the alignment between token embeddings and the global concept vector as this objective. We introduce one key modification: attribution is computed relative to the mean

Table 2: **SuperActivators yield more accurate and faithful attributions than global concept vectors.** Accuracy is measured by attribution F_1 (alignment with ground-truth masks), while faithfulness is measured by insertion scores (↑ is better) and deletion scores (↓ is better). This table shows results for *CLIP-ViT-L/14* linear separators on *COCO* and *Gemma-2-9B* linear separators on *iSarcasm*. Appendix K.2 demonstrates that these same trends hold across all other datasets, models, and concept vector types.

Attribution Method	Dataset	Attribution F_1 (↑ is better)		Insertion Score (↑ is better)		Deletion Score (↓ is better)	
		Concept	Super Activators	Concept	Super Activators	Concept	Super Activators
LIME [56]	COCO	0.29±0.02	**0.40±0.03**	0.333±0.009	**0.367±0.008**	0.010±0.001	**0.007±0.001**
	iSarcasm	0.76±0.02	**0.89±0.01**	0.383±0.008	**0.412±0.009**	0.009±0.000	**0.005±0.004**
SHAP [57]	COCO	0.35±0.01	**0.37±0.02**	0.334±0.004	**0.365±0.004**	0.010±0.001	**0.008±0.002**
	iSarcasm	0.77±0.03	**0.90±0.02**	0.384±0.008	**0.410±0.003**	0.009±0.001	**0.006±0.001**
RISE [58]	COCO	0.35±0.02	**0.38±0.03**	0.328±0.004	**0.354±0.007**	0.012±0.002	**0.009±0.000**
	iSarcasm	0.81±0.01	**0.94±0.03**	0.382±0.005	**0.409±0.009**	0.008±0.001	**0.005±0.002**
SHAP IQ [59]	COCO	0.34±0.01	**0.37±0.01**	0.330±0.005	**0.358±0.008**	0.011±0.002	**0.009±0.001**
	iSarcasm	0.79±0.02	**0.92±0.01**	0.379±0.004	**0.407±0.004**	0.009±0.001	**0.006±0.001**
IntGrad [60]	COCO	0.28±0.00	**0.35±0.04**	0.326±0.003	**0.359±0.005**	0.013±0.003	**0.010±0.003**
	iSarcasm	0.72±0.02	**0.84±0.01**	0.375±0.004	**0.405±0.009**	0.011±0.001	**0.008±0.003**
GradCAM [61]	COCO	0.37±0.01	**0.38±0.02**	0.329±0.005	**0.352±0.004**	0.012±0.003	**0.010±0.001**
	iSarcasm	0.74±0.02	**0.87±0.03**	0.377±0.004	**0.403±0.008**	0.010±0.001	**0.007±0.001**
FullGrad [62]	COCO	**0.43±0.01**	**0.43±0.00**	0.331±0.006	**0.357±0.010**	0.011±0.001	**0.009±0.002**
	iSarcasm	0.73±0.03	**0.85±0.01**	0.376±0.005	**0.402±0.010**	0.010±0.001	**0.007±0.001**
CALM [63]	COCO	**0.42±0.01**	**0.42±0.01**	0.332±0.010	**0.360±0.004**	0.011±0.002	**0.008±0.000**
	iSarcasm	0.78±0.01	**0.91±0.02**	0.380±0.007	**0.408±0.004**	0.009±0.001	**0.006±0.001**
MFABA [62]	COCO	0.33±0.01	**0.39±0.03**	0.339±0.005	**0.374±0.006**	0.006±0.001	**0.004±0.001**
	iSarcasm	0.77±0.02	**0.90±0.03**	0.391±0.002	**0.420±0.009**	0.006±0.001	**0.003±0.001**

embedding of local SuperActivators. Each SuperActivators is defined using the sparsity level δ that achieves the highest detection F_1 score on the validation set. For each concept c, attribution scores are then binarized into c-positive or c-negative using the threshold that maximizes validation F_1. If a sample contains no SuperActivators associated with concept c, all tokens are assigned as c-negative.

Our SuperActivator-based approach produces attribution maps that align more closely with ground-truth segmentation masks than those derived from global concept vectors. Across attribution methods, local SuperActivators consistently yield higher F_1 scores, outperforming the global baseline on both COCO and iSARCASM (Table 2), with similar improvements observed across additional image and text datasets, models, and concept vector types (Tables 3–9). Moreover, SuperActivators-based attributions achieve higher insertion and lower deletion scores, indicating greater faithfulness to model behavior (Table 2). Compared to global concept vectors, SuperActivators lead to faster convergence toward human-annotated cues, and removing such tokens results in a sharper drop in alignment, further highlighting their explanatory relevance.

Figures 6 and 35 illustrate that SuperActivators produce attribution maps that more accurately localize ground-truth concept regions. Tokens identified as important by SuperActivators align more closely with human annotations, whereas global concept vector attributions often highlight diffuse or semantically irrelevant areas.

5 Related Work

Concept-Based Interpretability: Concept-based interpretability techniques seek to link model internals with human-understandable features. Common approaches include defining concept vectors as linear separators (e.g., TCAV; [7]), or as centroid embeddings from labeled examples [50]. Unsupervised discovery methods include ACE [51], hierarchical clustering [34], matrix factorization approaches [64, 65], and sparse autoencoders [66, 67]. Across these works, concepts are assumed to be recoverable as structured vectors, clusters, or basis elements within representation space.

Challenges in Concept Representations: Many open questions remain concerning the structure of concept representations. The linearity hypothesis posits that concepts correspond to directions in activation space, linearly separable and recoverable with simple probes [68, 69]. Empirically, however, activations are often *entangled*, firing on tokens or samples where the concept is absent or bleeding into related but unintended semantics [20, 18], *polysemantic*, where a single neuron or direction encodes multiple features [19, 70], and *unstable*, with concept signals shifting across layers, spatial locations, exemplar sets, and random seeds [22, 71, 21, 72]. These properties can amplify failure modes such as spurious correlations [4] and concept leakage [73], undermining both detection and attribution. In response, some approaches try to modify model training to enforce more interpretable or disentangled concept structures [74, 75] or enforce structure (such as compositionality) in concepts extracted from pretrained models [76]. Our work takes a different perspective: rather than redesigning representations, we identify a sparse and reliable signal that already exists within otherwise noisy activation distributions.

Concept Detection: Concept detection is a central task in concept-based interpretability [22], with practical importance wherever one wants to determine whether a given concept is present in a sample—for example, detecting clinical or radiological concepts in medical images and reports [23, 24] or identifying undesirable online behavior [13, 16]. Most approaches instantiate a concept as a vector (e.g., a prototype or separator) and then score a sample by its alignment to that vector. This can be done using a *global* representation—such as the [CLS] token or pooled embeddings—which can be effective but often dilute sparse, fine-grained signals [47, 48]. When token or patch embeddings are available, methods instead compute token-level activations and aggregate them into a single alignment score; common choices include [CLS]-based scoring [16, 25, 77], mean pooling [26, 27, 12], max pooling [28, 22, 78, 35], or last-token scoring [29, 28, 48]. Beyond vector scoring, *concept bottleneck models* implicitly encode detection within a supervised concept layer designed for downstream tasks [79]. More recently, high-performing vision–language models have enabled *zero-shot prompting* that bypasses explicit concept vectors altogether, with strong results from CLIP and newer multimodal LMs (e.g., GPT-4o-mini) [22, 52, 28].

Feature Attributions for Concepts: Feature attributions for a given concept tells us *where* a concept is located within a sample [30]. Traditional attribution methods such as Integrated Gradients [60] and Grad-CAM [61], along with concept-based adaptations [7, 30, 25, 65], have been used to connect predictions to attribute concept alignment to input tokens. Beyond these, various works generate localization maps via direct alignment with raw activation scores [27, 78, 80, 78] and attention values [81].

6 Discussion and Future Work

In this work, we introduced and characterized the SuperActivator Mechanism, demonstrating that transformers concentrate reliable concept evidence into a sparse set of highly activated tokens. Leveraging this property enabled us to cut through the noise of globally aggregated concept vector activations and uncover more reliable signals of concept presence, which in turn serve as a stronger basis for concept localization. In the future, investigating how SuperActivators arise during training may provide deeper insight into how this mechanism emerges. Moreover, applying these principles in real-world settings for improved concept detection and localization offers the potential to make model interpretability more actionable in practice.

Acknowledgments

This research was partially supported by a gift from AWS AI to Penn Engineering's ASSET Center for Trustworthy AI, by ARPA-H program on Safe and Explainable AI under the award D24AC00253-00, by NSF award SLES 2331783, and by an NSF Graduate Research Fellowship under award DGE-2236662.

We acknowledge the use of large language models (ChatGPT, Gemini, and Claude Code) to assist with text drafting and editing, as well as code generation and debugging. All content was reviewed and revised by the authors to ensure accuracy and clarity.

References

[1] Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. Large language models: A survey, 2025. URL https://arxiv.org/abs/2402.06196.

[2] Ryan Greenblatt, Carson Denison, Benjamin Wright, Fabien Roger, Monte MacDiarmid, Sam Marks, Johannes Treutlein, Tim Belonax, Jack Chen, David Duvenaud, Akbir Khan, Julian Michael, Sören Mindermann, Ethan Perez, Linda Petrini, Jonathan Uesato, Jared Kaplan, Buck Shlegeris, Samuel R. Bowman, and Evan Hubinger. Alignment faking in large language models, 2024. URL https://arxiv.org/abs/2412.14093.

[3] Kevin Roose. A conversation with bing's chatbot left me deeply unsettled. *The New York Times*. URL https://www.nytimes.com/2023/02/16/technology/bing-sydney-microsoft-ai-chatbot.html.

[4] Yuhang Zhou, Paiheng Xu, Xiaoyu Liu, Bang An, Wei Ai, and Furong Huang. Explore spurious correlations at the concept level in language models for text classification, 2024. URL https://arxiv.org/abs/2311.08648.

[5] Yifan Yang, Xiaoyu Liu, Qiao Jin, Furong Huang, and Zhiyong Lu. Unmasking and quantifying racial bias of large language models in medical report generation. *Communications Medicine*, 4(1), September 2024. ISSN 2730-664X. doi: 10.1038/s43856-024-00601-z. URL http://dx.doi.org/10.1038/s43856-024-00601-z.

[6] Lukas Berglund, Meg Tong, Max Kaufmann, Mikita Balesni, Asa Cooper Stickland, Tomasz Korbak, and Owain Evans. The reversal curse: Llms trained on "a is b" fail to learn "b is a", 2024. URL https://arxiv.org/abs/2309.12288.

[7] 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*, pages 2668–2677. PMLR, 2018.

[8] Bolei Zhou, Yiyou Sun, David Bau, and Antonio Torralba. Interpretable basis decomposition for visual explanation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 119–134, 2018.

[9] Abubakar Abid, Mert Yuksekgonul, and James Zou. Meaningfully debugging model mistakes using conceptual counterfactual explanations, 2022. URL https://arxiv.org/abs/2106.12723.

[10] Chih-Kuan Yeh, Been Kim, Sercan Arik, Chun-Liang Li, Tomas Pfister, and Pradeep Ravikumar. On completeness-aware concept-based explanations in deep neural networks. *Advances in neural information processing systems*, 33:20554–20565, 2020.

[11] Nina Rimsky, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Matt Turner. Steering llama 2 via contrastive activation addition. *ArXiv*, abs/2312.06681, 2023. URL https://api.semanticscholar.org/CorpusID:266174252.

[12] Praneet Suresh, Jack Stanley, Sonia Joseph, Luca Scimeca, and Danilo Bzdok. From noise to narrative: Tracing the origins of hallucinations in transformers, 2025. URL https://arxiv.org/abs/2509.06938.

[13] Sheng Liu, Haotian Ye, Lei Xing, and James Y. Zou. In-context vectors: Making in context learning more effective and controllable through latent space steering. *ArXiv*, abs/2311.06668, 2023. URL https://api.semanticscholar.org/CorpusID:265149781.

[14] Zhihao Xu, Ruixuan Huang, Changyu Chen, and Xiting Wang. Uncovering safety risks of large language models through concept activation vector, 2024. URL https://arxiv.org/abs/2404.12038.

[15] Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J. Vazquez, Ulisse Mini, and Monte MacDiarmid. Steering language models with activation engineering, 2024. URL https://arxiv.org/abs/2308.10248.

[16] Isar Nejadgholi, Esma Balkır, Kathleen C. Fraser, and Svetlana Kiritchenko. Towards procedural fairness: Uncovering biases in how a toxic language classifier uses sentiment information, 2022. URL https://arxiv.org/abs/2210.10689.

[17] Jack Lindsey, Wes Gurnee, Emmanuel Ameisen, Brian Chen, Adam Pearce, Nicholas L. Turner, Craig Citro, David Abrahams, Shan Carter, Basil Hosmer, Jonathan Marcus, Michael Sklar, Adly Templeton, Trenton Bricken, Callum McDougall, Hoagy Cunningham, Thomas Henighan, Adam Jermyn, Andy Jones, Andrew Persic, Zhenyi Qi, T. Ben Thompson, Sam Zimmerman, Kelley Rivoire, Thomas Conerly, Chris Olah, and Joshua Batson. On the biology of a large language model. *Transformer Circuits Thread*, 2025. URL https://transformer-circuits.pub/2025/attribution-graphs/biology.html.

[18] Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. Zoom in: An introduction to circuits. *Distill*, 2020. doi: 10.23915/distill.00024.001. https://distill.pub/2020/circuits/zoom-in.

[19] Trenton Bricken, Adly Templeton, Jonathan Batson, Brian Chen, Adam Jermyn, Tom Conerly, and *et al.* Towards monosemanticity: Decomposing language models with dictionary learning. Anthropic Research Preprint, 2023. Available at Anthropic's website.

[20] Gabriel Goh, Nick Cammarata †, Chelsea Voss †, Shan Carter, Michael Petrov, Ludwig Schubert, Alec Radford, and Chris Olah. Multimodal neurons in artificial neural networks. *Distill*, 2021. doi: 10.23915/distill.00030. https://distill.pub/2021/multimodal-neurons.

[21] Angus Nicolson, Lisa Schut, J. Alison Noble, and Yarin Gal. Explaining explainability: Recommendations for effective use of concept activation vectors, 2025. URL https://arxiv.org/abs/2404.03713.

[22] Zhengxuan Wu, Aryaman Arora, Atticus Geiger, Zheng Wang, Jing Huang, Dan Jurafsky, Christopher D. Manning, and Christopher Potts. Axbench: Steering llms? even simple baselines outperform sparse autoencoders, 2025. URL https://arxiv.org/abs/2501.17148.

[23] Johannes Rückert, Asma Ben Abacha, Alba Garcia Seco de Herrera, Louise Bloch, Raphael Brüngel, Ahmad Idrissi-Yaghir, Henning Schäfer, Henning Müller, and Christoph M. Friedrich. Overview of imageclefmedical 2023 – caption prediction and concept detection. In *CLEF 2023: Conference and Labs of the Evaluation Forum*, September 2023.

[24] Tudor Groza, Harrison Caufield, Daniel Gration, et al. An evaluation of gpt models for phenotype concept recognition. *BMC Medical Informatics and Decision Making*, 24(30), 2024. doi: 10.1186/s12911-024-02439-w. URL https://doi.org/10.1186/s12911-024-02439-w.

[25] Xuemin Yu, Fahim Dalvi, Nadir Durrani, and Hassan Sajjad. Latent concept-based explanation of nlp models. *ArXiv*, abs/2404.12545, 2024. URL https://api.semanticscholar.org/CorpusID:269282778.

[26] Alex McKenzie, Urja Pawar, Phil Blandfort, William Bankes, David Krueger, Ekdeep Singh Lubana, and Dmitrii Krasheninnikov. Detecting high-stakes interactions with activation probes. *ArXiv*, abs/2506.10805, 2025. URL https://api.semanticscholar.org/CorpusID:279318482.

[27] Itay Benou and Tammy Riklin-Raviv. Show and tell: Visually explainable deep neural nets via spatially-aware concept bottleneck models, 2025. URL https://arxiv.org/abs/2502.20134.

[28] Henk Tillman and Dan Mossing. Investigating task-specific prompts and sparse autoencoders for activation monitoring, 2025. URL https://arxiv.org/abs/2504.20271.

[29] Runjin Chen, Andy Arditi, Henry Sleight, Owain Evans, and Jack Lindsey. Persona vectors: Monitoring and controlling character traits in language models. *ArXiv*, abs/2507.21509, 2025. URL https://api.semanticscholar.org/CorpusID:280337840.

[30] Antonio De Santis, Riccardo Campi, Matteo Bianchi, and Marco Brambilla. Visual-tcav: Concept-based attribution and saliency maps for post-hoc explainability in image classification. *ArXiv*, abs/2411.05698, 2024. URL https://api.semanticscholar.org/CorpusID:273950563.

[31] Yang Zhang, Yawei Li, Hannah Brown, Mina Rezaei, Bernd Bischl, Philip Torr, Ashkan Khakzar, and Kenji Kawaguchi. Attributionlab: Faithfulness of feature attribution under controllable environments. *arXiv preprint arXiv:2310.06514*, 2023. URL https://arxiv.org/abs/2310.06514.

[32] Kaitlyn Zhou, Kawin Ethayarajh, and Dan Jurafsky. Frequency-based distortions in contextualized word embeddings, 2021. URL https://arxiv.org/abs/2104.08465.

[33] Baturay Saglam, Paul Kassianik, Blaine Nelson, Sajana Weerawardhena, Yaron Singer, and Amin Karbasi. Large language models encode semantics in low-dimensional linear subspaces, 2025. URL https://arxiv.org/abs/2507.09709.

[34] Fahim Dalvi, Abdul Rafae Khan, Firoj Alam, Nadir Durrani, Jia Xu, and Hassan Sajjad. Discovering latent concepts learned in bert, 2022. URL https://arxiv.org/abs/2205.07237.

[35] Yan Xie, Zequn Zeng, Hao Zhang, Yucheng Ding, Yi Wang, Zhengjue Wang, Bo Chen, and Hongwei Liu. Discovering fine-grained visual-concept relations by disentangled optimal transport concept bottleneck models, 2025. URL https://arxiv.org/abs/2505.07209.

[36] Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C. Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[37] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision (ECCV)*, 2014.

[38] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge, 2010.

[39] Sean Bell, Paul Upchurch, Noah Snavely, and Kavita Bala. Opensurfaces: A richly annotated catalog of surface appearance. *ACM Transactions on Graphics (SIGGRAPH)*, 32(4), 2013.

[40] David Bau, Jun-Yan Zhu, Hendrik Strobelt, Agata Lapedriza, Bolei Zhou, and Antonio Torralba. Understanding the role of individual units in a deep neural network. *Proceedings of the National Academy of Sciences*, 2020. ISSN 0027-8424. doi: 10.1073/pnas.1907375117. URL https://www.pnas.org/content/early/2020/08/31/1907375117.

[41] Silviu Oprea and Walid Magdy. isarcasm: A dataset of intended sarcasm. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2020.

[42] Dorottya Demszky, Dana Movshovitz-Attias, Jeongwoo Ko, Alan Cowen, Gaurav Nemade, and Sujith Ravi. Goemotions: A dataset of fine-grained emotions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 4040–4054, 2020.

[43] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021. URL https://arxiv.org/abs/2103.00020.

[44] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[45] Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, Johan Ferret, Peter Liu, Pouya Tafti, Abe Friesen, Michelle Casbon, Sabela Ramos, Ravin Kumar, Charline Le Lan, Sammy Jerome, Anton Tsitsulin, Nino Vieillard, Piotr Stanczyk, Sertan Girgin, Nikola Momchev, Matt Hoffman, Shantanu Thakoor, Jean-Bastien Grill, Behnam Neyshabur, Olivier Bachem, Alanna Walton, Aliaksei Severyn, Alicia Parrish, Aliya Ahmad, Allen Hutchison, Alvin Abdagic, Amanda Carl, Amy Shen, Andy Brock, Andy Coenen, Anthony Laforge, Antonia Paterson, Ben Bastian, Bilal Piot, Bo Wu, Brandon Royal, Charlie Chen, Chintu Kumar, Chris Perry, Chris Welty, Christopher A. Choquette-Choo, Danila Sinopalnikov, David Weinberger, Dimple Vijaykumar, Dominika Rogozińska, Dustin Herbison, Elisa Bandy, Emma Wang, Eric Noland, Erica Moreira, Evan Senter, Evgenii Eltyshev, Francesco Visin, Gabriel Rasskin, Gary Wei, Glenn Cameron, Gus Martins, Hadi Hashemi, Hanna Klimczak-Plucińska, Harleen Batra, Harsh Dhand, Ivan Nardini, Jacinda Mein, Jack Zhou, James Svensson, Jeff Stanway, Jetha Chan, Jin Peng Zhou, Joana Carrasqueira, Joana Iljazi, Jocelyn Becker, Joe Fernandez, Joost van Amersfoort, Josh Gordon, Josh Lipschultz, Josh Newlan, Ju yeong Ji, Kareem Mohamed, Kartikeya Badola, Kat Black, Katie Millican, Keelin McDonell, Kelvin Nguyen, Kiranbir Sodhia, Kish Greene, Lars Lowe Sjoesund, Lauren Usui, Laurent Sifre, Lena Heuermann, Leticia Lago, Lilly McNealus, Livio Baldini Soares, Logan Kilpatrick, Lucas Dixon, Luciano Martins, Machel Reid, Manvinder Singh, Mark Iverson, Martin Görner, Mat Velloso, Mateo Wirth, Matt Davidow, Matt Miller, Matthew Rahtz, Matthew Watson, Meg Risdal, Mehran Kazemi, Michael Moynihan, Ming Zhang, Minsuk Kahng, Minwoo Park, Mofi Rahman, Mohit Khatwani, Natalie Dao, Nenshad Bardoliwalla, Nesh Devanathan, Neta Dumai, Nilay Chauhan, Oscar Wahltinez, Pankil Botarda, Parker Barnes, Paul Barham, Paul Michel, Pengchong Jin, Petko Georgiev, Phil Culliton, Pradeep Kuppala, Ramona Comanescu, Ramona Merhej, Reena Jana, Reza Ardeshir Rokni, Rishabh Agarwal, Ryan Mullins, Samaneh Saadat, Sara Mc Carthy, Sarah Cogan, Sarah Perrin, Sébastien M. R. Arnold, Sebastian Krause, Shengyang Dai, Shruti Garg, Shruti Sheth, Sue Ronstrom, Susan Chan, Timothy Jordan, Ting Yu, Tom Eccles, Tom Hennigan, Tomas Kocisky, Tulsee Doshi, Vihan Jain, Vikas Yadav, Vilobh Meshram, Vishal Dharmadhikari, Warren Barkley, Wei Wei, Wenming Ye, Woohyun Han, Woosuk Kwon, Xiang Xu, Zhe Shen, Zhitao Gong, Zichuan Wei, Victor Cotruta, Phoebe Kirk, Anand Rao, Minh Giang, Ludovic Peran, Tris Warkentin, Eli Collins, Joelle Barral, Zoubin Ghahramani, Raia Hadsell, D. Sculley, Jeanine Banks, Anca Dragan, Slav Petrov, Oriol Vinyals, Jeff Dean, Demis Hassabis, Koray Kavukcuoglu, Clement Farabet, Elena Buchatskaya, Sebastian Borgeaud, Noah Fiedel, Armand Joulin, Kathleen Kenealy, Robert Dadashi, and Alek Andreev. Gemma 2: Improving open language models at a practical size, 2024. URL `https://arxiv.org/abs/2408.00118`.

[46] Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. Qwen3 embedding: Advancing text embedding and reranking through foundation models, 2025. URL `https://arxiv.org/abs/2506.05176`.

[47] Hyunjin Choi, Judong Kim, Seongho Joe, and Youngjune Gwon. Evaluation of bert and albert sentence embedding performance on downstream nlp tasks, 2021. URL `https://arxiv.org/abs/2101.10642`.

[48] Yixuan Tang and Yi Yang. Pooling and attention: What are effective designs for llm-based embedding models?, 2024. URL `https://arxiv.org/abs/2409.02727`.

[49] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ArXiv*, abs/2010.11929, 2020. URL `https://api.semanticscholar.org/CorpusID:225039882`.

[50] Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, Shashwat Goel, Nathaniel Li, Michael J. Byun, Zifan Wang, Alex Troy Mallen, Steven Basart, Sanmi Koyejo, Dawn Song, Matt Fredrikson, Zico Kolter, and Dan Hendrycks. Representation engineering: A top-down approach to ai transparency. *ArXiv*, abs/2310.01405, 2023.

[51] Amirata Ghorbani, James Wexler, and Been Kim. Automating interpretability: Discovering and testing visual concepts learned by neural networks. *ArXiv*, abs/1902.03129, 2019. URL https://api.semanticscholar.org/CorpusID:59842921.

[52] Peter Robicheaux, Matvei Popov, Anish Madan, Isaac Robinson, Joseph Nelson, Deva Ramanan, and Neehar Peri. Roboflow100-vl: A multi-domain object detection benchmark for vision-language models. *ArXiv*, abs/2505.20612, 2025. URL https://api.semanticscholar.org/CorpusID:278910603.

[53] Teresa Dorszewski, Lenka Tvetkov'a, Robert Jenssen, Lars Kai Hansen, and Kristoffer Wickstrøm. From colors to classes: Emergence of concepts in vision transformers. *ArXiv*, abs/2503.24071, 2025. URL https://api.semanticscholar.org/CorpusID:277467666.

[54] Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes, 2018. URL https://arxiv.org/abs/1610.01644.

[55] David Arps, Younes Samih, Laura Kallmeyer, and Hassan Sajjad. Probing for constituency structure in neural language models. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 6738–6757, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-emnlp.502. URL https://aclanthology.org/2022.findings-emnlp.502/.

[56] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should i trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1135–1144, 2016.

[57] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in neural information processing systems 30*, 2017.

[58] Vitali Petsiuk, Abir Das, and Kate Saenko. Rise: Randomized input sampling for explanation of black-box models. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2018.

[59] Thomas Fel, Alexandre Jullien, David Vigouroux, Remi Cadene, Thomas Nicodeme, Matthieu Laly, Asma Fermanian, Benjamin Audit, and Thomas Scantamburlo. Explaining groups of instances with shap-iq. In *International Conference on Artificial Intelligence and Statistics*, pages 6467–6491. PMLR, 2023.

[60] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International conference on machine learning*, pages 3319–3328. PMLR, 2017.

[61] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017.

[62] Suraj Srinivas and François Fleuret. Full-gradient representation for neural network visualization. In *Advances in Neural Information Processing Systems 32*, 2019.

[63] Divyanshu Mahajan, Chenhao Tan, and Matthew Turek. Calm: A causality-guided framework for generating local and global model explanations. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1215–1224, 2021.

[64] Lihua Zhang and Shihua Zhang. A unified joint matrix factorization framework for data integration. *ArXiv*, abs/1707.08183, 2017. URL https://api.semanticscholar.org/CorpusID:21228616.

[65] Thomas Fel, Agustin Picard, Louis Béthune, Thibaut Boissin, David Vigouroux, Julien Colin, Rémi Cadène, and Thomas Serre. Craft: Concept recursive activation factorization for explainability. *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2711–2721, 2022. URL https://api.semanticscholar.org/CorpusID:253708233.

[66] Hoagy Cunningham, Aidan Ewart, Logan Riggs, Robert Huben, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. *ArXiv*, abs/2309.08600, 2023. URL https://api.semanticscholar.org/CorpusID:261934663.

[67] Leo Gao, Tom Dupr'e la Tour, Henk Tillman, Gabriel Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan Leike, and Jeffrey Wu. Scaling and evaluating sparse autoencoders. *ArXiv*, abs/2406.04093, 2024. URL https://api.semanticscholar.org/CorpusID:270286001.

[68] Tomas Mikolov, Wen tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *North American Chapter of the Association for Computational Linguistics*, 2013. URL https://api.semanticscholar.org/CorpusID:7478738.

[69] Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg, and Christopher Olah. Toy models of superposition, 2022. URL https://arxiv.org/abs/2209.10652.

[70] Laura O'Mahony, Vincent Andrearczyk, Henning Müller, and Mara Graziani. Disentangling neuron representations with concept vectors. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 3770–3775, 2023. doi: 10.1109/CVPRW59228.2023.00390.

[71] Anita Mahinpei, Justin Clark, Isaac Lage, Finale Doshi-Velez, and Weiwei Pan. Promises and pitfalls of black-box concept learning models. *ArXiv*, abs/2106.13314, 2021. URL https://api.semanticscholar.org/CorpusID:235652059.

[72] Georgii Mikriukov, Gesina Schwalbe, Christian Hellert, and Korinna Bade. *Evaluating the Stability of Semantic Concept Representations in CNNs for Robust Explainability*, page 499–524. Springer Nature Switzerland, 2023. ISBN 9783031440670. doi: 10.1007/978-3-031-44067-0_26. URL http://dx.doi.org/10.1007/978-3-031-44067-0_26.

[73] Enrico Parisini, Tapabrata Chakraborti, Chris Harbron, Ben D. MacArthur, and Christopher R. S. Banerji. Leakage and interpretability in concept-based models, 2025. URL https://arxiv.org/abs/2504.14094.

[74] Zhi Chen, Yijie Bei, and Cynthia Rudin. Concept whitening for interpretable image recognition. *Nature Machine Intelligence*, 2:772 – 782, 2020. URL https://api.semanticscholar.org/CorpusID:211031886.

[75] Xin Wang, Hong Chen, Si'ao Tang, Zihao Wu, and Wenwu Zhu. Disentangled representation learning, 2024. URL https://arxiv.org/abs/2211.11695.

[76] Adam Stein, Aaditya Naik, Yinjun Wu, Mayur Naik, and Eric Wong. Towards compositionality in concept learning. *ArXiv*, abs/2406.18534, 2024.

[77] Maike Behrendt, Stefan Sylvius Wagner, and Stefan Harmeling. Maxpoolbert: Enhancing bert classification via layer- and token-wise aggregation. *ArXiv*, abs/2505.15696, 2025. URL https://api.semanticscholar.org/CorpusID:278782887.

[78] Hyesu Lim, Jinho Choi, Jaegul Choo, and Steffen Schneider. Sparse autoencoders reveal selective remapping of visual concepts during adaptation, 2025. URL https://arxiv.org/abs/2412.05276.

[79] Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept bottleneck models. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 5338–5348. PMLR, 13–18 Jul 2020.

[80] Hong Zhou, Rui Zhang, Peifeng Lai, Chaoran Guo, Yong Wang, Zhida Sun, and Junjie Li. El-vit: Probing vision transformer with interactive visualization, 2024. URL https://arxiv.org/abs/2401.12666.

[81] Yossi Gandelsman, Alexei A. Efros, and Jacob Steinhardt. Interpreting clip's image representation via text-based decomposition. *ArXiv*, abs/2310.05916, 2023. URL https://api.semanticscholar.org/CorpusID:263829688.

[82] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Conference on Empirical Methods in Natural Language Processing*, 2019. URL https://api.semanticscholar.org/CorpusID:201646309.

[83] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547, 2019.

[84] OpenAI. Gpt-4o system card, 2024. URL https://openai.com/index/gpt-4o-system-card/. Model documentation and safety evaluation.

[85] Zara Siddique, Liam D. Turner, and Luis Espinosa-Anke. Dialz: A python toolkit for steering vectors, 2025. URL https://arxiv.org/abs/2505.06262.

[86] Zhiyu Zhu, Huaming Chen, Jiayu Zhang, Xinyi Wang, Zhibo Jin, Minhui Xue, Dongxiao Zhu, and Kim-Kwang Raymond Choo. Mfaba: A more faithful and accelerated boundary-based attribution method for deep neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(15):17228–17236, 2024. doi: 10.1609/aaai.v38i15.29669.

[87] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade W Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, Patrick Schramowski, Srivatsa R Kundurthy, Katherine Crowson, Ludwig Schmidt, Robert Kaczmarczyk, and Jenia Jitsev. LAION-5b: An open large-scale dataset for training next generation image-text models. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022. URL https://openreview.net/forum?id=M3Y74vmsMcY.

[88] Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. Openclip, July 2021. URL https://doi.org/10.5281/zenodo.5143773. If you use this software, please cite it as below.

[89] Tom Lieberum, Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Nicolas Sonnerat, Vikrant Varma, János Kramár, Anca Dragan, Rohin Shah, and Neel Nanda. Gemma scope: Open sparse autoencoders everywhere all at once on gemma 2, 2024. URL https://arxiv.org/abs/2408.05147.

[90] Bartosz Cywiński and Kamil Deja. Saeuron: Interpretable concept unlearning in diffusion models with sparse autoencoders, 2025. URL https://arxiv.org/abs/2501.18052.

[91] Leo Gao, Tom Dupré la Tour, Henk Tillman, Gabriel Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan Leike, and Jeffrey Wu. Scaling and evaluating sparse autoencoders, 2024. URL https://arxiv.org/abs/2406.04093.

A SuperActivator Visual Examples

This section presents visual examples of SuperActivators in test samples across multiple image and text datasets. The heatmaps illustrate the activation score between the token embeddings and the labeled concept vectors, where red indicates high alignment, blue indicates low alignment, and a green rectangle indicates SuperActivators. The concepts used in these visualizations are linear separators trained on *LLaMA-3.2-11B-Vision-Instruct* embeddings at the model depth that achieved the highest validation performance, with SuperActivators defined at the sparsity level δ that yielded the best validation F_1 for each concept.

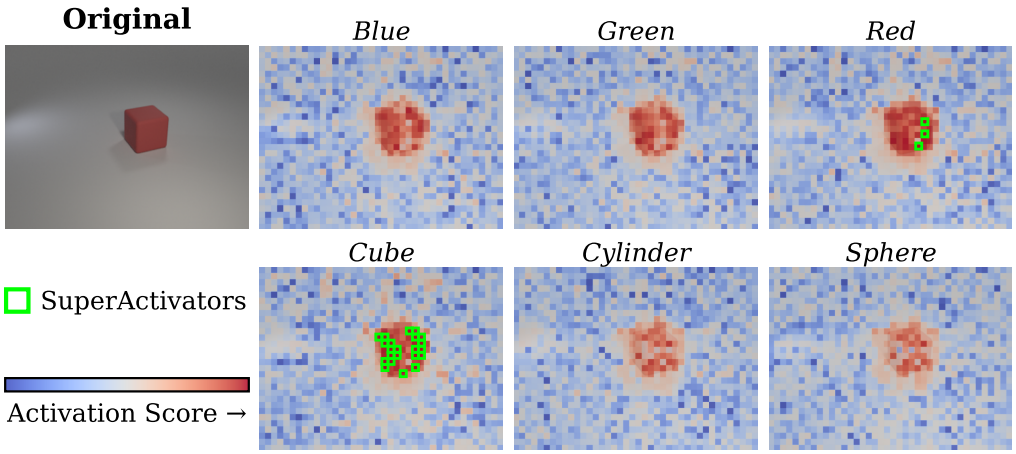


Figure 7: *CLEVR* – Visualization of Concept Activations and SuperActivators

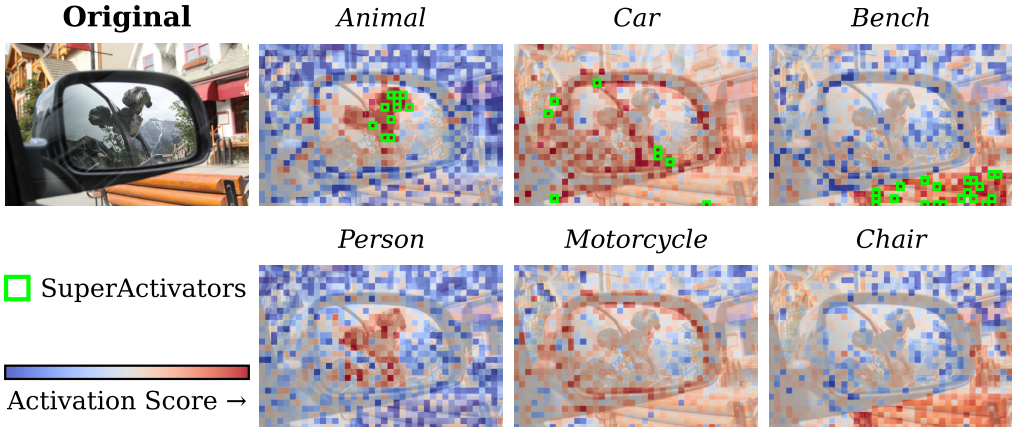


Figure 8: *COCO* – Visualization of Concept Activations and SuperActivators

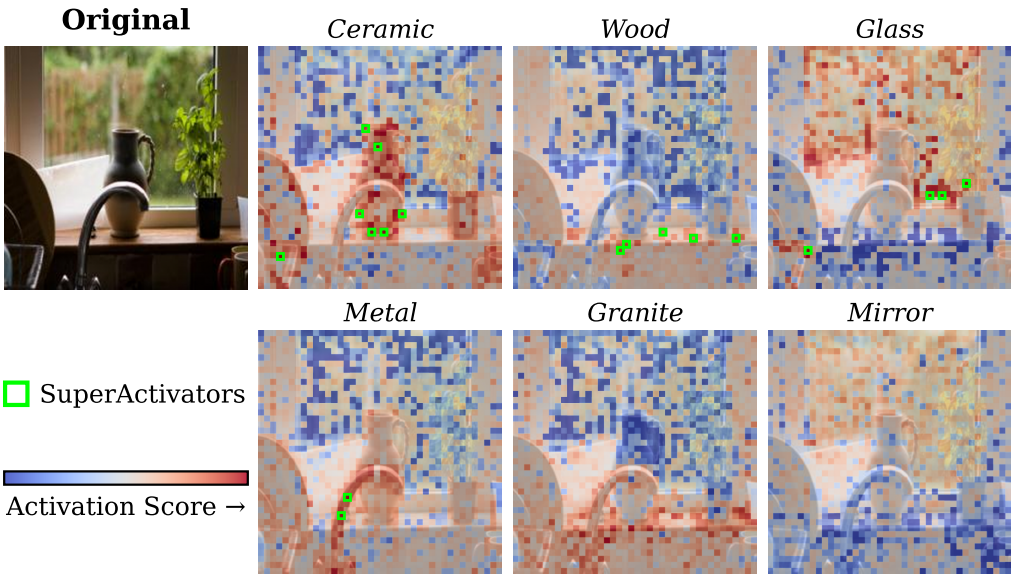


Figure 9: *OpenSurfaces* – Visualization of Concept Activations and SuperActivators

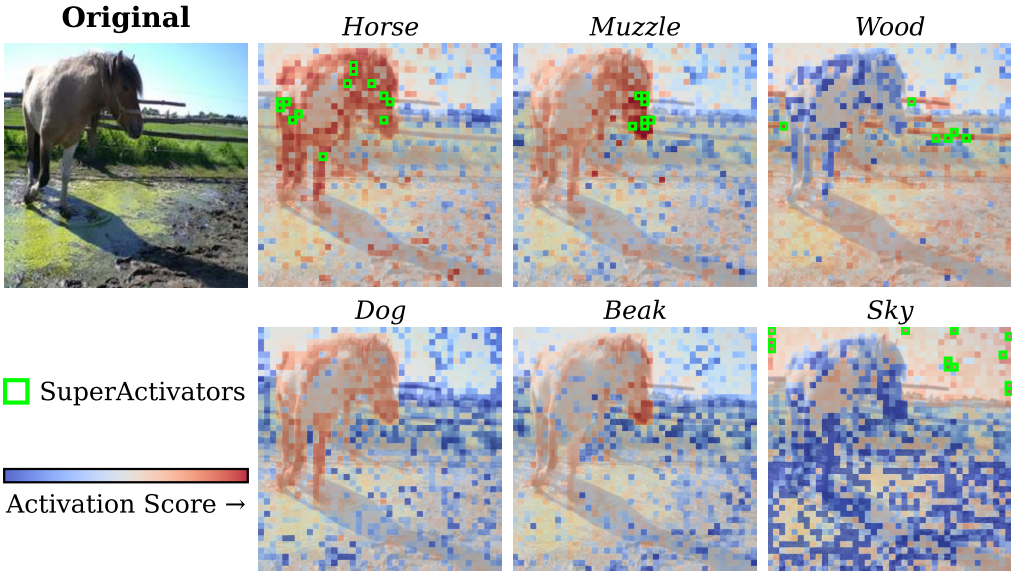


Figure 10: *Pascal* – Visualization of Concept Activations and SuperActivators

Original Text (No Labeled Concept):

Regrettably, my morning coffee spilled all over my fresh white shirt. I was running late for work and in my rush, I knocked my coffee mug right off the counter. Thankfully, I had a spare shirt in my car.

Sarcasm Activations:

Regrettably, my morning coffee spilled all over my fresh white shirt. I was running late for work and in my rush, I knocked my coffee mug right off the counter. Thankfully, I had a spare shirt in my car.

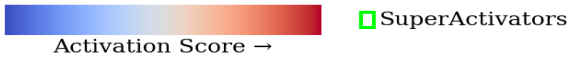
(a) Non-Sarcastic Version

Original Text (*Sarcasm* highlighted):

It's such a treat when my morning coffee decides to spill all over my fresh white shirt. I was running late for work and in my rush, I knocked my coffee mug right off the counter. Thankfully, I had a spare shirt in my car.

Sarcasm Activations:

It's such a treat when my morning coffee decides to spill all over my fresh white shirt. I was running late for work and in my rush, I knocked my coffee mug right off the counter. Thankfully, I had a spare shirt in my car.



(b) Sarcastic Version

Figure 11: *Sarcasm* – Visualization of Concept Activations and SuperActivators (sarcastic and non-sarcastic version of same sentiment)

Original Text (No Labeled Concept):

the worst way to wake up is when the alarm is too loud. it makes me feel really startled first thing in the morning. #NeedCoffee

Sarcastic Activations:

the worst way to wake up is when the alarm is too loud. it makes me feel really startled first thing in the morning. #NeedCoffee

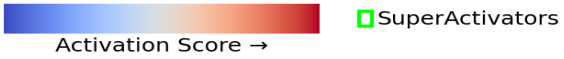
(a) Non-Sarcastic Sample

Original Text (*Sarcastic* highlighted):

there's no better way to wake up than having one dog jump directly on your stomach and knock the wind out of you while the other drop a dead rodent on the end of the bed. i really need to start closing the bedroom door at night. #morningchaos

Sarcastic Activations:

there's no better way to wake up than having one dog jump directly on your stomach and knock the wind out of you while the other drop a dead rodent on the end of the bed. i really need to start closing the bedroom door at night. #morningchaos



(b) Sarcastic Sample

Figure 12: *Sarcasm* – Visualization of Concept Activations and SuperActivators (non-sarcastic and sarcastic text samples)

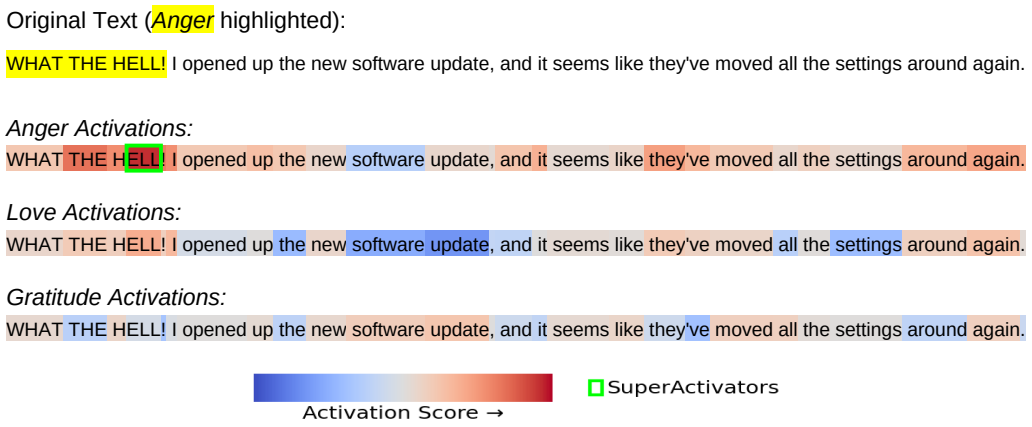
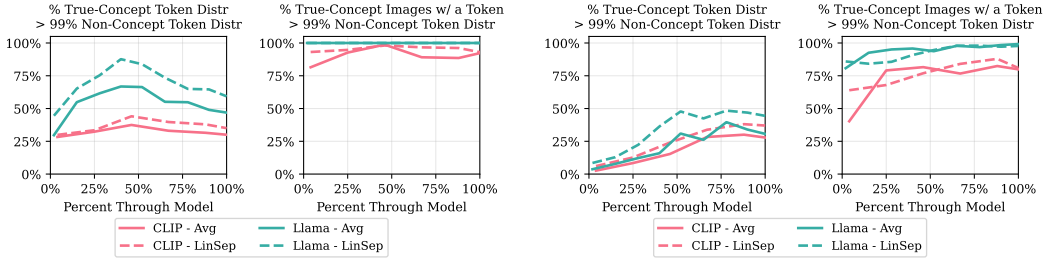


Figure 13: *Augmented GoEmotions* SuperActivator Example

B Motivation for Focusing on SuperActivators

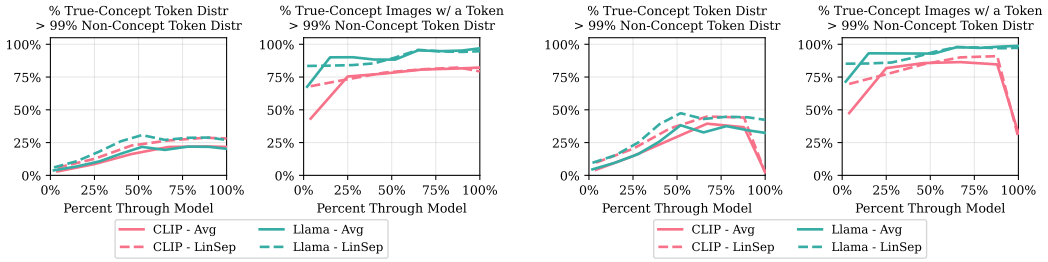
In this section, we motivate our focus on the highly-aligned activations in the tail of the in-concept activation distribution, $\mathcal{D}_c^{\text{in}}$. For this initial inquiry, we consider a token separable from the empirical out-of-concept activation distribution D_c^{out} if its concept activation is greater than 99% of the out-of-concept token activations, $q_{0.99}(D_c^{\text{out}})$. Then, for each dataset, on the left we plot the percent of in-concept token activations that are separable from out-of-concept activations (averaged across concepts) as a function of model depth. On the right, we plot the percentage of in-concept samples (images, comments, tweets, etc) that contain at least one token that is separable from the out-of-concept distribution as a function of model depth (again, averaged across concepts). In Figure 14, we report results across various datasets and models, as well as both average and linear separator concept vectors.

Generally, as shown in the leftmost plots, the percentage of well-separated in-concept token activations gradually increases throughout the model. However, the majority of the in-concept token activations typically do not exceed $q_{0.99}(D_c^{\text{out}})$ even at the most distinguishing layers, indicating a fundamental problem with separability. This problem is particularly severe for the text datasets. For the image concepts, most of the true-concept images have at least one well-separated token activation, and this separation generally also increases with model depth. In the text setting, while not all in-concept samples contain an activated patch, a substantial proportion do—indicating that some concept signal is present, albeit more diffuse. This likely reflects the specific text datasets used here, where concepts such as sarcasm and emotion are more subjective and nuanced than the object and texture annotations in image data. The main takeaway from these results is that across all image and text datasets, models, and concept types, there appears to be activations in the tail of D_c^{in} that are well-separated from D_c^{in} and carry signals of concept presence.



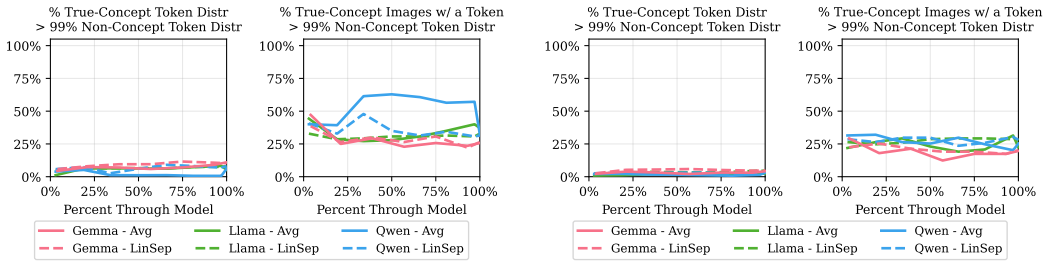
(a) CLEVR

(b) COCO



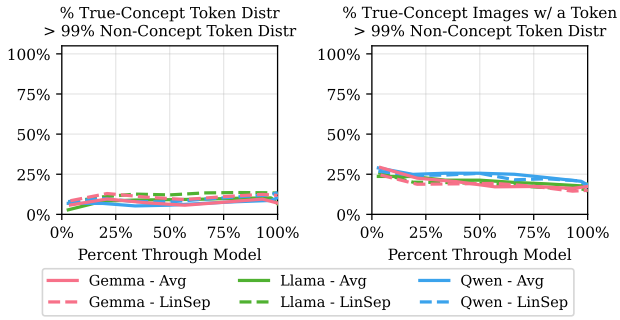
(c) OpenSurfaces

(d) Pascal



(e) Sarcasm

(f) iSarcasm



(g) GoEmotions

Figure 14: Across all image and text datasets, models, and concept types, there appears to be high magnitude in-concept activations that are well-separated from D_c^{in} and carry signals of concept presence.

C Experimental Configurations

C.1 Embedding Models

For images, we extract both input token and [CLS] token embeddings from the *CLIP* ViT-L/14 [43] and *LLaMA-3.2-11B-Vision-Instruct* (Meta, 2024) models. For text, we use *LLaMA-3.2-11B-Vision-Instruct*, *Gemma-2-9B* [45], and *Qwen3-Embedding-4B* [46]. Since these text models lack an explicit [CLS] token, we approximate a [CLS]-style representation by averaging token embeddings [47–49, 82]. For each model, we obtain embeddings across multiple layers. To ensure comparability, we normalize and mean-center each layer's embeddings using statistics computed from the training set.

To make the computation feasible, we evaluate models at a fixed set of percentage depths through the network, rather than at every layer. The chosen checkpoint percentages are *CLIP*: [4, 25, 46, 67, 88, 100], *LLaMA-Vision*: [2, 15, 28, 40, 52, 65, 78, 90, 100], *LLaMA-Text*: [3, 19, 34, 50, 66, 81, 97, 100], *Gemma*: [4, 21, 39, 57, 75, 93, 100], *Qwen*: [3, 19, 34, 50, 66, 81, 97, 100].

C.2 Concept Extraction Methods

Throughout, let x denote a sample (image or text), and $z(x) \in \mathbb{R}^d$ its embedding obtained from the underlying model. For a ground-truth concept c, let $\mathcal{X}_c^+$ denote the set of samples labeled positive for c. We use $v_c \in \mathbb{R}^d$ to denote the concept vector associated with c, and v_j to denote candidate concept vectors discovered by an unsupervised method. All concepts are constructed only using embeddings from the training set.

We extract concepts using supervised methods, unsupervised methods, and a prompting baseline. Concept representations are computed at both the token level, using embeddings from input tokens, and the [CLS] level, using embeddings from the [CLS] tokens, which lie in a distinct representational space optimized for sequence-level summarization.

Supervised Methods:

1. **Mean Prototypes** [50]: Each concept vector is defined as the average embedding of all positive examples,
$$v_c = \frac{1}{|\mathcal{X}_c^+|} \sum_{x \in \mathcal{X}_c^+} z(x).$$

2. **Linear Separators (LinSep)** [7]: For each concept c, we train a linear model (without bias) to distinguish positives from negatives. For training, we balance positive and negative samples and use `BCEWithLogitsLoss` with the Adam optimizer (learning rate 0.01). We train for up to 100 epochs with a batch size of 32, apply weight decay of $1\text{e}{-4}$, and decay the learning rate by a factor of 0.5 every 10 epochs. Early stopping is used with a patience of 15 epochs and a tolerance of 3, which sets the minimum improvement required to continue training. The resulting normal vector of the separating hyperplane is used as the concept vector:
$$v_c = w_c.$$

Unsupervised Methods:

1. **K-Means Prototypes** [51, 34]: We cluster embeddings using FAISS GPU [83] with Euclidean distance, a maximum of 300 iterations, and k=1000 for token-level embeddings and k=50 for [CLS] embeddings. The choice of k was determined experimentally using an elbow curve. Token-level embeddings are finer-grained and therefore benefit from a larger number of clusters. Each cluster centroid is used as a concept vector:
$$v_j = \mu_j = \frac{1}{|\mathcal{C}_j|} \sum_{x \in \mathcal{C}_j} z(x).$$

2. **Cluster-Based Separators (K-LinSep)**: We first assign soft labels to embeddings based on their K-means cluster membership, then train linear separators with the same procedure

described above to predict whether an embedding belongs to a given cluster. The normal vectors of these separators are treated as concept directions:

$$v_{ij} = w_{ij}.$$

3. **Sparse Autoencoders (SAEs)** [19]: SAEs learn a sparse reconstruction

$$z(x) \approx Wh(x), \quad h(x) \in \mathbb{R}^m \text{ sparse}, \quad v_j = w_j,$$

where each column w_j of W corresponds to a candidate concept. Because SAE training is computationally expensive, we use pretrained SAEs; see Appendix L for architectural and implementation details.

To ensure we can evaluate against unsupervised methods, each ground-truth concept c is matched to the unsupervised unit v_j that achieves the highest validation F_1 score for detecting c:

$$v_c = \arg\max_{v_j} \ F_1^{\text{val}}(c, v_j).$$

Prompt Baseline: As a non-concept vector baseline, we query *LLaMA-3.2-11B-Vision-Instruct* directly. For each sample x and concept c, we prompt:

"Is the concept of c present in the following? x".

Prior works have employed similar zero-shot prompting baselines successfully [22, 52, 28].

C.3 Dataset Overview

CLEVR (Single-Object) [36]: A synthetic dataset of 1,000 images, each containing a red, green, or blue object with shape sphere, cylinder, or cube. Images and segmentation masks are generated programmatically, allowing fine-grained control over object properties and patch-level annotations.

COCO [37]: We use the 2017 validation set of *MS-Coco*, containing 5,500 images with everyday scenes involving people, objects, and natural contexts. Each image comes with human-annotated segmentations, providing dense labels for both object categories and broader supercategories.

Broden–Pascal [38] and Broden–OpenSurfaces [39]: We use 4,503 samples from Pascal and 3,578 samples from OpenSurfaces. These are subsets of the Broden dataset [40], which unifies multiple segmentation datasets into a single benchmark for concept-based interpretability research. Pascal primarily contains natural images with segmented objects from diverse categories such as animals, vehicles, and household items, while OpenSurfaces emphasizes fine-grained material and surface property annotations (e.g., wood, fabric, metal). We chose these two subsets because they focus on patch-level segmentation where concepts do not necessarily span the entire image.

Sarcasm (Fully Synthetic): We generate a dataset of 1,446 paragraphs, where roughly half contain exactly one sarcastic sentence surrounded by neutral sentences.

iSarcasm (Augmented): We adapt 1,734 samples from the original iSarcasm dataset [41], which provides sarcastic tweets alongside non-sarcastic rewrites conveying the same meaning (both provided by the original authors). We augment these by embedding sarcastic and non-sarcastic sentences into short paragraphs of neutral context, with sarcastic spans explicitly marked.

GoEmotions (Augmented): We use 5,427 samples from the GoEmotions dataset [42], a human-annotated collection of Reddit comments labeled with 27 emotion categories. We augment selected samples by embedding emotional sentences within surrounding neutral context, tagging the emotional span while preserving natural paragraph flow.

C.4 Text Augmentation Pipelines and Prompts

This section describes the augmentation pipelines used for generating and adapting text datasets, along with the exact prompts. Our goal was to create datasets with localized token-level concept spans, since most publicly available text datasets only provide sample-level (sentence, tweet, comment, etc) labels. Generation and augmentation are performed via controlled prompting of GPT-4o [84].

C.4.1 Sarcasm (Fully Synthetic)

Pipeline: We generate entirely new paragraphs containing exactly one sarcastic sentence. The sarcastic sentence is wrapped in <SARCASM> tags, while all other sentences are neutral. This ensures that each paragraph contains exactly one labeled sarcastic span, with natural context surrounding it. By constraining sarcastic content to a single line, we obtain a controlled setup where token-level supervision is precise and unambiguous.

Prompt:

```
Write 10 short paragraphs (4-8 sentences each). Each paragraph must include
**exactly one sarcastic sentence**, wrapped in <SARCASM> ... </SARCASM> tags.

Guidelines:
- The sarcastic sentence should be subtle, deadpan, or context-dependent.
- All other sentences must be sincere and literal.
- Vary topic, tone, and structure across paragraphs.

Only the sarcastic line may be wrapped in tags.

Return only the 10 numbered paragraphs.
```

Example: Jane always prided herself on her cooking abilities. <SARCASM>Indeed, the local fire department must have also appreciated her culinary exploits, given the number of times they've had to rush to her house.</SARCASM> Still, she was not deterred and continued to experiment in the kitchen, determined to perfect her skills. She understood that learning anything new involved a process of trial and error.

C.4.2 iSarcasm Augmentation

Dataset Overview: The original iSarcasm dataset contains sarcastic tweets paired with author-provided sincere rewrites conveying the same meaning. We extend this dataset synthetically by surrounding the sarcastic tweets with literal, neutral context, ensuring precise span-level supervision. Only sarcastic samples are selected for augmentation, and for each sarcastic input we generate both a sarcastic augmented post and a non-sarcastic rewrite.

Augmentation Pipeline: Each sarcastic input is expanded into casual, paragraph-like text using controlled prompting of GPT-4.0. To introduce variation, random structural features are applied:

- 20% chance of forcing a `[Sarcasm][Trigger]` structure.
- 15% chance of adding emojis or hashtags.
- Otherwise, a random choice among `[Sarcasm][Trigger]`, `[Trigger][Sarcasm]`, or `[Trigger][Sarcasm][Trigger]`.

Sarcastic Augmentation Prompt:

```
You are a data annotation machine. Your only goal is to produce perfectly literal
text that follows the rules. You must not be creative or clever. You must not
generate any figurative language outside of the provided tags.

Your Task:
You will be given a sarcastic tweet and its true meaning. Rewrite the tweet by
embedding it within a strictly literal train of thought that matches the original's
casual tone.

Structure: [Randomly choose or force specific structure]
[Optional emoji/hashtag instruction if selected]

Constraints Checklist:
```

```
- The tone is casual and informal.
- The added text is not redundant.
- Outside <SARCASM> tags is strictly literal and descriptive.
- The original sarcastic tweet is fully preserved within <SARCASM> tags.
- Output contains ONLY the final post.

Input Sarcastic Tweet: "{sarcastic_tweet}"
Sincere Meaning (for your context): "{rephrased_text}"

Your Output:
```

Non-Sarcastic Augmentation Prompt.

```
You are a data annotation machine. Your only goal is to produce perfectly literal
text that follows the rules. You must not be creative or clever. You must not
invent new details.

Your Task:
Take a sincere idea and expand it slightly into a personal, casual post,
remaining 100% faithful to the original meaning.

[Optional emoji/hashtag instruction if selected]

Constraints Checklist:
- The tone is casual and informal.
- The entire post is strictly literal and descriptive.
- No sarcasm, irony, overstatement, or rhetorical questions.
- The post must be 100% faithful to the meaning of the original idea.
- Output contains ONLY the final post.

Input Sincere Idea: "{rephrased_text}"

Your Output:
```

Verification Process: Outputs are verified via flexible matching with progressively lenient checks: exact matching (case-insensitive), whitespace normalization, URL/punctuation removal, and word-overlap thresholds. If all attempts fail, the original tweet is wrapped in <SARCASM> tags as a fallback.

Example:

> **Input sarcastic tweet:** "The only thing I got from college is a caffeine addiction."
> **Input sincere rephrase:** "College is really difficult, expensive, tiring, and I often question if a degree is worth the stress."
>
> **Sarcastic augmentation:** "I just checked my calendar and saw how many assignments are due this week. <SARCASM>the only thing i got from college is a caffeine addiction</SARCASM>"
> **Non-sarcastic rewrite:** "college is really difficult. it's also expensive and tiring. sometimes i find myself questioning if getting a degree is worth all the stress."

C.4.3 GoEmotions Augmentation

Dataset Overview: GoEmotions is a large-scale dataset of Reddit comments labeled with up to 27 fine-grained emotions. We extend it synthetically by surrounding the original emotional comment with strictly neutral filler context, ensuring the emotional span remains localized and clearly marked with <EMOTION> tags.

Augmentation Pipeline: Every comment in GoEmotions is augmented without filtering, following a two-step process:

1. **Step 1: Generation.** A "Neutral Filler Machine" prompt is used to generate five diverse neutral-context options embedding the original emotional comment.
2. **Step 2: Selection.** A "Grader" prompt evaluates the five drafts and selects the best single option according to neutrality and naturalness.

To increase variation, a random structure is sampled per comment:

- 50% chance: `[Emotion][Context]`
- 25% chance: `[Context][Emotion]`
- 25% chance: `[Context][Emotion][Context]`

Step 1 — Neutral Filler Prompt:

```
You are a Neutral Filler Machine. Your task is to generate neutral,
non-emotional text to surround a given Reddit comment.

Task:
- Preserve the original emotional comment exactly inside <EMOTION> tags.
- Generate five unique and diverse neutral contexts that flow naturally.
- All options must follow the required structure.

Constraints:
- Text outside <EMOTION> must be strictly neutral (no emotion leakage).
- Sound natural and casual like a Reddit post.
- No redundancy with the emotional comment.

Input Emotional Comment: "{emotional_comment}"
Primary Emotion(s): "{emotion_labels_str}"
Required Structure: "{structure_choice}"

Your Output: Five options, each in the correct structure.
```

Step 2 — Selection Prompt.

```
You are a data annotation quality assurance specialist.
Your task is to select the best draft among five options.

Checklist:
- Context must be strictly neutral (no emotions).
- Flow naturally as a Reddit comment.
- No contradiction or redundancy.
- Only output the single best final option.

Draft Options:
{draft_options}

Your Final, Best Output:
```

Verification Process: The augmented comments are verified using flexible string matching to ensure that the original text is preserved inside `<EMOTION>` tags. We allow up to five retry attempts with progressively lenient checks. If all attempts fail, the fallback is to wrap the original comment directly in `<EMOTION>` tags.

Example:

> **Original emotional comment (gratitude):** "I didn't know that, thank you for teaching me something today!"

Augmented output: "A comment explained the process behind recycling plastics and how it affects the environment. <EMOTION>I didn't know that, thank you for teaching me something today!</EMOTION>"

C.5 Concepts Used in Experiments

For the MS-COCO, GoEmotions, and Broden datasets, we filter concepts using minimum sample thresholds (100–300 samples, depending on the dataset) to ensure sufficient data for reliable concept construction, though future work could examine SuperActivators in underfit settings. The semantic concepts used in our experiments are listed here:

- **CLEVR:** blue, green, red, cube, cylinder, sphere
- **COCO:** accessory, animal, appliance, bench, book, bottle, bowl, bus, car, chair, couch, cup, dining table, electronic, food, furniture, indoor, kitchen, motorcycle, outdoor, person, pizza, potted plant, sports, train, truck, tv, umbrella, vehicle
- **Broden–OpenSurfaces:** brick, cardboard, carpet, ceramic, concrete, fabric, food, fur, glass, granite, hair, laminate, leather, metal, mirror, painted, paper, plastic-clear, plastic-opaque, rock, rubber, skin, tile, wallpaper, wicker, wood
- **Broden–Pascal:** airplane, bicycle, bird, boat, body, book, building, bus, cap, car, cat, cup, dog, door, ear, engine, grass, hair, horse, leg, mirror, motorbike, mountain, painting, person, pottedplant, saddle, screen, sky, sofa, table, track, train, tvmonitor, wheel, wood, arm, bag, beak, bottle, box, cabinet, ceiling, chain wheel, chair, coach, curtain, eye, eyebrow, fabric, fence, floor, foot, ground, hand, handle bar, head, headlight, light, mouth, muzzle, neck, nose, paw, plant, plate, plaything, pole, pot, road, rock, rope, shelves, sidewalk, signboard, stern, tail, torso, tree, wall, water, windowpane, wing
- **Sarcasm:** sarcasm.
- **iSarcasm:** sarcastic.
- **GoEmotions:** confusion, joy, sadness, anger, love, caring, optimism, amusement, curiosity, disapproval, approval, annoyance, gratitude, admiration

D Concept Formalisms in More Detail

We provide a detailed formalization of concept detection and activation aggregation strategies, focusing on transformer architectures given their demonstrated effectiveness across modalities.

Model Representations. Let f be a trained transformer model that processes an input $x \in \mathcal{X}$ (an image or a text sequence) into a set of hidden representations. At a given layer ℓ, we extract token-level embeddings

$$f_\ell(x) = \{\, z_1^{\text{tok}}(x), \ldots, z_{n(x)}^{\text{tok}}(x), z^{\text{cls}}(x) \,\}, \quad z_i^{\text{tok}}(x), z^{\text{cls}}(x) \in \mathbb{R}^d.$$

Here $z_i^{\text{tok}}(x)$ denotes the representation of the i-th token (or image patch), and $z^{\text{cls}}(x)$ denotes the [CLS]-style representation summarizing the full input.

Concept Vectors and Activation Scores. For any semantic concept c, we define a **concept vector** $v_c \in \mathbb{R}^d$, extracted via one of the techniques in Appendix C.2. Intuitively, v_c represents a direction in embedding space along which the concept c is encoded. The **activation score** of an embedding z with respect to concept c is defined as

$$s_c(z) = \langle z, v_c \rangle.$$

If v_c is derived as a cluster centroid, this corresponds to cosine similarity (for normalized embeddings). If v_c is derived from a linear separator, it corresponds to the signed distance from the separating hyperplane. Intuitively, $s_c(z)$ measures the alignment of z with concept c: large positive values indicate that z strongly encodes features associated with c, while negative values suggest opposition or absence.

We aim to characterize, for each concept c, the distribution of activation scores across many samples. Let $\mathcal{D}_c^{\text{in}}$ and $\mathcal{D}_c^{\text{out}}$ denote the population-level distributions of activation scores for in-concept and

out-of-concept tokens, respectively. Empirically, we approximate these distributions using finite datasets D_c^{in} and D_c^{out} constructed from observed activations. Let Z denote the set of all tokens across samples, and let $S_c = \{ s_c(z) : z \in Z \}$ be their corresponding activation scores. If $Z_c^{\text{in}} \subseteq Z$ are the tokens labeled concept-positive for concept c and Z_c^{out} are the tokens drawn from samples that do *not* contain c (thus excluding out-of-concept tokens from samples containing c to avoid self-attention leakage), then

$$D_c^{\text{in}} = \{ s_c(z) : z \in Z_c^{\text{in}} \}, \qquad D_c^{\text{out}} = \{ s_c(z) : z \in Z_c^{\text{out}} \},$$

which serve as empirical samples from $\mathcal{D}_c^{\text{in}}$ and $\mathcal{D}_c^{\text{out}}$. We use $Q_q(\mathcal{D})$ to denote the population q-quantile of a distribution $\mathcal{D}$, and $q_q(D)$ to denote its empirical estimate computed from a finite sample D.

Concept Detection. The goal of concept detection is to determine whether a sample x contains a concept c [22]. Transformer models produce a collection of activation scores at the token level, but for detection we require a single score per sample. This necessitates an **aggregation operator** that interprets the set of token-level activations as a sample-level score.

Let $S_c(x) = \{s_{c,1}(x), \ldots, s_{c,n(x)}(x), s_{c,\text{cls}}(x)\}$ denote the set of activation scores for concept c on input x, where $s_{c,i}(x)$ is the score for the i-th token and $s_{c,\text{cls}}(x)$ is the score for the [CLS] token. An aggregation operator is any function

$$G : \mathbb{R}^{n(x)+1} \to \mathbb{R}, \quad s_c^{\text{agg}}(x) = G(S_c(x)).$$

Given a calibrated threshold τ_c, detection is performed by

$$\hat{y}_c(x) = \mathbf{1}[\, s_c^{\text{agg}}(x) \geq \tau_c \,].$$

Because prior work has shown that different concepts may emerge at different layers of a transformer [33, 25, 34], we calibrate the layer separately for each concept to avoid enforcing a strict shared choice. This calibration is also performed independently for each aggregation strategy, ensuring that no operator is unfairly advantaged or disadvantaged due to layer-specific biases.

Standard Aggregation Strategies. Prior work has considered several choices of G, each operating on the same token-level activations (with the exception of [CLS], which uses separately trained concept vectors since sample-level and input token-level representations occupy different spaces):

- **[CLS]-only (G_{cls}):**
$$G_{\text{cls}}(S_c(x)) = s_{c,\text{cls}}(x).$$
Uses only the [CLS] token score. Since CLS tokens are trained to attend to all inputs, they are natural candidates for summarizing sample-level concepts, and this strategy has been widely adopted [16, 25, 77].
- **Mean pooling (G_{mean}):**
$$G_{\text{mean}}(S_c(x)) = \tfrac{1}{n(x)} \sum_{i=1}^{n(x)} s_{c,i}(x).$$
Averages over all tokens. This ensures that no part of the input is ignored and can capture distributed concept signals, a technique used in multiple studies [27, 12, 85].
- **Max pooling (G_{max}):**
$$G_{\text{max}}(S_c(x)) = \max\{s_{c,1}(x), \ldots, s_{c,n(x)}(x), s_{c,\text{cls}}(x)\}.$$
Takes the strongest activation across input tokens. This is effective for isolating the most distinct concept signals [28, 22, 78, 35].
- **Last token (G_{last}):**
$$G_{\text{last}}(S_c(x)) = s_{c,n(x)}(x).$$
Uses the last input token activation. For autoregressive models, the final token often encodes sequence-level information, making it a plausible summary for concept detection [29, 28, 48].

- **Random token (G_{rand}):**

$$G_{\text{rand}}(S_c(x)) = s_{c,j}(x), \quad j \sim \text{Unif}\{1, \ldots, n(x)\}.$$

  Selects an input token activation uniformly at random. While a weak baseline, self-attention mechanisms distribute information broadly, so even a randomly chosen token may retain meaningful concept cues.

These operators differ only in how they interpret activations; they do not alter how concept vectors are trained. Thresholds τ_c are determined using a validation set (e.g., from a fixed grid of percentiles), and detection at test time is performed by applying the same G to the sample activations and comparing against τ_c.

SuperActivator Aggregation. We develop an aggregation strategy that takes advantage of the SuperActivators mechanism we identified, using the highest-activation tokens in the global true-concept distribution as the basis for concept detection.

Formally, let

$$\mathcal{S}^{+}_{\text{val},c} \;=\; \left\{\, s_{c,i}(x) \;\middle|\; x \in \mathcal{X}^{+}_{\text{val},c},\; i \in \{1, \ldots, n(x)\} \right\}$$

be the set of all token-level activations for c from validation samples where c is present. For a chosen percentage δ (selected from a fixed grid), we define the *SuperActivator threshold* as

$$\tau_c^{\text{super}} \;=\; Q_{1-\delta}\big(\mathcal{S}^{+}_{\text{val},c}\big),$$

so that only the top δ percent of in-concept activations exceed τ_c^{super}. Unlike traditional max pooling approaches, which calibrate thresholds based on the single maximum activation per sample, our approach looks at the highest activations generally in the in-concept distribution, allowing us to consider multiple high-fidelity token activations per sample where calibrating.

At test time, we aggregate using a max operator,

$$G_{\text{super}}(S_c(x)) \;=\; \max S_c(x),$$

and predict presence if this maximum exceeds the calibrated SuperActivator threshold:

$$\hat{y}_c^{\text{super}}(x) \;=\; \mathbf{1}[\, G_{\text{super}}(S_c(x)) \;\geq\; \tau_c^{\text{super}} \,].$$

δ is calibrated per concept on the validation set to maximize detection F_1. Beyond providing thresholds for reporting overall detection scores, this calibration also allows us to analyze how varying the sparsity level of the SuperActivator mechanism impacts performance.

E Comprehensive Concept Detection Results

The following tables compare our SuperActivator-based detection method with baseline approaches across all datasets, models, and concept types. Each table reports the average F_1 detection scores, computed as the mean across concepts weighted by their frequency in the test set. In each table, the top-performing concept detection method for each model/concept type combination is in **bold** and the second best-performing is underlined.

On the image datasets (i.e., *CLEVR*, *MS-Coco*, *OpenSurfaces*, and *Pascal*), our SuperActivator method consistently outperforms all other concept detection methods, except for a couple instances in the very simple *CLEVR* dataset, where prompting achieves the highest performance by a small margin. Though sometimes the CLS-based achieves near-equivalent performance, zero-shot prompting is most consistently the next best detection method. For the text datasets, (i.e., *Sarcasm*, *Augmented iSarcasm*, and *Augmented GoEmotions*), our SuperActivator also achieves consistently high detection performance across configurations. However, particularly for the *Augmented iSarcasm* dataset, CLS-based methods are able to outperform our SuperActivator, though usually by a very small amount that falls within the margin of error.

Overall, these results confirm that across image and text modalities, model families, and concept types, SuperActivator tokens provide a highly reliable signal of concept presence.

*Concept detection F_1 for the **CLEVR** dataset.*

Model	Concept Type	Concept Detection Methods					
		RandTok	LastTok	MeanTok	CLS	Prompt	SuperAct (Ours)
CLIP	Avg	0.526 ± 0.028	0.542 ± 0.027	0.684 ± 0.020	0.957 ± 0.017	**0.987 ± 0.009**	0.986 ± 0.009
	Linsep	0.745 ± 0.009	0.706 ± 0.008	0.840 ± 0.009	0.963 ± 0.015	0.987 ± 0.009	**0.991 ± 0.007**
	K-Means	0.727 ± 0.013	0.878 ± 0.016	0.976 ± 0.013	0.959 ± 0.016	0.987 ± 0.009	**0.991 ± 0.007**
	K-Linsep	0.737 ± 0.017	0.848 ± 0.017	0.907 ± 0.019	0.965 ± 0.015	**0.987 ± 0.009**	0.950 ± 0.015
Llama	Avg	0.645 ± 0.018	0.591 ± 0.019	0.660 ± 0.018	0.955 ± 0.017	0.987 ± 0.009	**0.998 ± 0.003**
	Linsep	0.967 ± 0.090	0.879 ± 0.004	0.920 ± 0.004	0.961 ± 0.015	0.987 ± 0.009	**0.997 ± 0.004**
	K-Means	0.775 ± 0.089	0.946 ± 0.090	0.955 ± 0.013	0.928 ± 0.021	**0.987 ± 0.009**	0.959 ± 0.013
	K-Linsep	0.717 ± 0.024	0.910 ± 0.016	0.910 ± 0.015	0.962 ± 0.015	0.987 ± 0.009	**0.989 ± 0.008**

*Concept detection F_1 for the **COCO** dataset.*

Model	Concept Type	Concept Detection Methods					
		RandTok	LastTok	MeanTok	CLS	Prompt	SuperAct (Ours)
CLIP	Avg	0.575 ± 0.012	0.503 ± 0.012	0.494 ± 0.013	0.685 ± 0.012	0.686 ± 0.050	**0.721 ± 0.012**
	Linsep	0.606 ± 0.011	0.687 ± 0.011	0.592 ± 0.011	0.702 ± 0.011	0.686 ± 0.050	**0.787 ± 0.011**
	K-Means	0.525 ± 0.013	0.517 ± 0.013	0.337 ± 0.012	0.583 ± 0.012	0.686 ± 0.050	**0.694 ± 0.012**
	K-Linsep	0.486 ± 0.012	0.523 ± 0.012	0.333 ± 0.011	0.571 ± 0.013	0.686 ± 0.050	**0.696 ± 0.012**
Llama	Avg	0.485 ± 0.011	0.457 ± 0.012	0.378 ± 0.012	0.534 ± 0.013	0.686 ± 0.050	**0.746 ± 0.012**
	Linsep	0.606 ± 0.011	0.680 ± 0.011	0.551 ± 0.011	0.566 ± 0.013	0.686 ± 0.050	**0.829 ± 0.010**
	K-Means	0.510 ± 0.012	0.491 ± 0.012	0.373 ± 0.011	0.447 ± 0.013	0.686 ± 0.050	**0.747 ± 0.011**
	K-Linsep	0.493 ± 0.011	0.477 ± 0.012	0.363 ± 0.011	0.430 ± 0.013	0.686 ± 0.050	**0.716 ± 0.011**

*Concept detection F_1 for the **OpenSurfaces** dataset.*

Model	Concept Type	Concept Detection Methods					
		RandTok	LastTok	MeanTok	CLS	Prompt	SuperAct (Ours)
CLIP	Avg	0.438 ± 0.014	0.419 ± 0.013	0.403 ± 0.014	0.484 ± 0.014	0.491 ± 0.063	**0.538 ± 0.014**
	Linsep	0.470 ± 0.014	0.470 ± 0.014	0.427 ± 0.014	0.492 ± 0.014	0.491 ± 0.063	**0.551 ± 0.014**
	K-Means	0.443 ± 0.015	0.441 ± 0.015	0.373 ± 0.013	0.444 ± 0.010	0.491 ± 0.063	**0.544 ± 0.014**
	K-Linsep	0.432 ± 0.013	0.454 ± 0.012	0.365 ± 0.011	0.443 ± 0.009	0.491 ± 0.063	**0.543 ± 0.012**
Llama	Avg	0.404 ± 0.012	0.375 ± 0.012	0.361 ± 0.012	0.446 ± 0.014	0.491 ± 0.063	**0.534 ± 0.014**
	Linsep	0.438 ± 0.014	0.410 ± 0.014	0.390 ± 0.014	0.456 ± 0.013	0.491 ± 0.063	**0.558 ± 0.015**
	K-Means	0.443 ± 0.010	0.431 ± 0.011	0.360 ± 0.010	0.423 ± 0.005	0.491 ± 0.063	**0.545 ± 0.009**
	K-Linsep	0.439 ± 0.010	0.416 ± 0.011	0.360 ± 0.010	0.409 ± 0.011	0.491 ± 0.063	**0.545 ± 0.008**

*Concept detection F_1 for the **Pascal** dataset.*

Model	Concept Type	Concept Detection Methods					
		RandTok	LastTok	MeanTok	CLS	Prompt	SuperAct (Ours)
CLIP	Avg	0.612 ± 0.006	0.546 ± 0.006	0.594 ± 0.006	0.721 ± 0.006	0.680 ± 0.048	**0.788 ± 0.006**
	Linsep	0.723 ± 0.005	0.674 ± 0.005	0.678 ± 0.005	0.740 ± 0.006	0.680 ± 0.048	**0.826 ± 0.005**
	K-Means	0.533 ± 0.005	0.623 ± 0.002	0.490 ± 0.005	0.652 ± 0.003	0.680 ± 0.048	**0.770 ± 0.001**
	K-Linsep	0.574 ± 0.005	0.577 ± 0.004	0.466 ± 0.005	0.633 ± 0.004	0.680 ± 0.048	**0.756 ± 0.002**
Llama	Avg	0.536 ± 0.006	0.510 ± 0.006	0.502 ± 0.006	0.619 ± 0.007	0.680 ± 0.048	**0.786 ± 0.006**
	Linsep	0.659 ± 0.006	0.602 ± 0.006	0.590 ± 0.006	0.645 ± 0.006	0.680 ± 0.048	**0.822 ± 0.005**
	K-Means	0.507 ± 0.006	0.601 ± 0.006	0.481 ± 0.006	0.568 ± 0.007	0.680 ± 0.048	**0.792 ± 0.005**
	K-Linsep	0.499 ± 0.006	0.550 ± 0.006	0.443 ± 0.006	0.558 ± 0.007	0.680 ± 0.048	**0.784 ± 0.006**

*Concept detection F_1 for the **Sarcasm** dataset.*

Model	Concept Type	Concept Detection Methods					
		RandTok	LastTok	MeanTok	CLS	Prompt	SuperAct (Ours)
Llama	Avg	0.659 ± 0.052	0.706 ± 0.051	0.659 ± 0.052	0.694 ± 0.060	0.679 ± 0.074	**0.818 ± 0.051**
	Linsep	0.659 ± 0.060	0.683 ± 0.048	0.659 ± 0.060	0.737 ± 0.055	0.679 ± 0.074	**0.870 ± 0.039**
	K-Means	0.659 ± 0.061	0.659 ± 0.061	0.659 ± 0.061	0.665 ± 0.053	0.679 ± 0.074	**0.818 ± 0.049**
	K-Linsep	0.659 ± 0.054	0.670 ± 0.050	0.659 ± 0.052	0.658 ± 0.053	0.679 ± 0.074	**0.826 ± 0.048**
Qwen	Avg	0.662 ± 0.055	0.659 ± 0.066	0.659 ± 0.066	**0.687 ± 0.055**	0.679 ± 0.074	0.679 ± 0.060
	Linsep	0.659 ± 0.055	0.662 ± 0.051	0.659 ± 0.055	0.750 ± 0.054	0.679 ± 0.074	**0.857 ± 0.046**
	K-Means	0.659 ± 0.054	0.659 ± 0.054	0.659 ± 0.054	0.640 ± 0.059	0.679 ± 0.074	**0.717 ± 0.062**
	K-Linsep	0.659 ± 0.054	0.716 ± 0.057	0.659 ± 0.054	0.675 ± 0.053	0.679 ± 0.074	**0.769 ± 0.057**
Gemma	Avg	0.659 ± 0.058	0.659 ± 0.058	0.659 ± 0.058	0.665 ± 0.059	0.679 ± 0.074	**0.727 ± 0.056**
	Linsep	0.659 ± 0.059	0.668 ± 0.051	0.670 ± 0.051	0.686 ± 0.057	0.679 ± 0.074	**0.810 ± 0.051**
	K-Means	0.659 ± 0.053	0.659 ± 0.053	0.659 ± 0.053	0.658 ± 0.053	**0.679 ± 0.074**	0.659 ± 0.052
	K-Linsep	0.659 ± 0.054	**0.682 ± 0.054**	0.659 ± 0.054	0.670 ± 0.053	0.679 ± 0.074	0.659 ± 0.052

*Concept detection F_1 for the **Augmented iSarcasm** dataset.*

Model	Concept Type	Concept Detection Methods					
		RandTok	LastTok	MeanTok	CLS	Prompt	SuperAct (Ours)
Llama	Avg	0.677 ± 0.043	0.676 ± 0.043	0.676 ± 0.043	**0.867 ± 0.038**	0.789 ± 0.047	0.818 ± 0.043
	Linsep	0.885 ± 0.035	0.717 ± 0.029	0.791 ± 0.029	0.912 ± 0.031	0.789 ± 0.047	**0.924 ± 0.029**
	K-Means	0.737 ± 0.048	0.677 ± 0.055	0.677 ± 0.055	**0.809 ± 0.041**	0.789 ± 0.047	0.787 ± 0.044
	K-Linsep	0.811 ± 0.038	0.828 ± 0.040	0.708 ± 0.045	0.802 ± 0.041	0.789 ± 0.047	**0.866 ± 0.038**
Qwen	Avg	0.676 ± 0.041	0.679 ± 0.041	0.678 ± 0.041	**0.890 ± 0.034**	0.789 ± 0.047	0.757 ± 0.041
	Linsep	0.814 ± 0.041	0.711 ± 0.038	0.739 ± 0.041	**0.917 ± 0.030**	0.789 ± 0.047	0.895 ± 0.034
	K-Means	0.676 ± 0.076	0.676 ± 0.076	0.676 ± 0.076	**0.856 ± 0.038**	0.789 ± 0.047	0.788 ± 0.046
	K-Linsep	0.749 ± 0.044	0.676 ± 0.043	0.676 ± 0.043	**0.878 ± 0.036**	0.789 ± 0.047	0.832 ± 0.042
Gemma	Avg	0.735 ± 0.045	0.686 ± 0.039	0.702 ± 0.045	**0.899 ± 0.032**	0.789 ± 0.047	0.839 ± 0.038
	Linsep	0.853 ± 0.031	0.789 ± 0.035	0.789 ± 0.035	**0.904 ± 0.033**	0.789 ± 0.047	0.892 ± 0.034
	K-Means	0.676 ± 0.073	0.676 ± 0.073	0.676 ± 0.044	**0.827 ± 0.040**	0.789 ± 0.047	0.810 ± 0.045
	K-Linsep	0.676 ± 0.043	0.679 ± 0.046	0.754 ± 0.043	**0.864 ± 0.038**	0.789 ± 0.047	0.825 ± 0.044

*Concept detection F_1 for the **Augmented GoEmotions** dataset.*

Model	Concept Type	Concept Detection Methods					
		RandTok	LastTok	MeanTok	CLS	Prompt	SuperAct (Ours)
Llama	Avg	0.293 ± 0.027	0.216 ± 0.027	0.216 ± 0.026	0.277 ± 0.028	0.252 ± 0.100	**0.383 ± 0.028**
	Linsep	0.372 ± 0.028	0.307 ± 0.027	0.193 ± 0.029	0.320 ± 0.029	0.252 ± 0.100	**0.459 ± 0.029**
	K-Means	0.305 ± 0.028	0.281 ± 0.029	0.117 ± 0.028	0.192 ± 0.022	0.252 ± 0.100	**0.417 ± 0.028**
	K-Linsep	0.426 ± 0.027	0.365 ± 0.027	0.327 ± 0.028	0.213 ± 0.022	0.252 ± 0.100	**0.448 ± 0.028**
Qwen	Avg	0.277 ± 0.026	0.214 ± 0.026	0.151 ± 0.026	0.347 ± 0.028	0.252 ± 0.100	**0.431 ± 0.027**
	Linsep	0.305 ± 0.028	0.248 ± 0.025	0.199 ± 0.026	0.357 ± 0.028	0.252 ± 0.100	**0.458 ± 0.027**
	K-Means	0.341 ± 0.028	0.284 ± 0.027	0.111 ± 0.026	0.192 ± 0.021	0.252 ± 0.100	**0.451 ± 0.027**
	K-Linsep	0.390 ± 0.026	0.373 ± 0.027	0.365 ± 0.026	0.191 ± 0.022	0.252 ± 0.100	**0.453 ± 0.028**
Gemma	Avg	0.336 ± 0.024	0.313 ± 0.023	0.151 ± 0.022	0.366 ± 0.029	0.252 ± 0.100	**0.394 ± 0.026**
	Linsep	0.352 ± 0.026	0.301 ± 0.026	0.190 ± 0.027	0.361 ± 0.029	0.252 ± 0.100	**0.420 ± 0.028**
	K-Means	0.294 ± 0.028	0.213 ± 0.025	0.132 ± 0.025	0.218 ± 0.020	0.252 ± 0.100	**0.422 ± 0.026**
	K-Linsep	0.339 ± 0.028	0.315 ± 0.024	0.360 ± 0.025	0.205 ± 0.019	0.252 ± 0.100	**0.414 ± 0.028**

F Ablation: How does concept detection performance vary with depth?

In this section, we investigate how average concept detection performance evolves throughout model depth. Figures 15 and 16 visualize heatmaps of the average detection F_1 scores as a function of transformer layer depth for image and text datasets, respectively. Each heatmap reports the mean F_1 score across all datasets for each model, concept type, and detection scheme, computed over a grid of model depths. These heatmaps help illustrate how concept signals emerge and strengthen at different stages within the network.

In the vision domain, detection performance generally increases with depth, plateauing around the middle layers and declining slightly at the final layer. This behavior aligns with findings from prior work [33, 25, 34], which report that mid-level and late-level layers often capture the richest and most separable semantic information. A similar trend can be observed in text-based models, though with greater variability across datasets and concept types. These results highlight that the most reliable concept signals tend to emerge most clearly past intermediate layers, and that SuperActivator-based detection consistently distinguishes concept presence better than baselines.

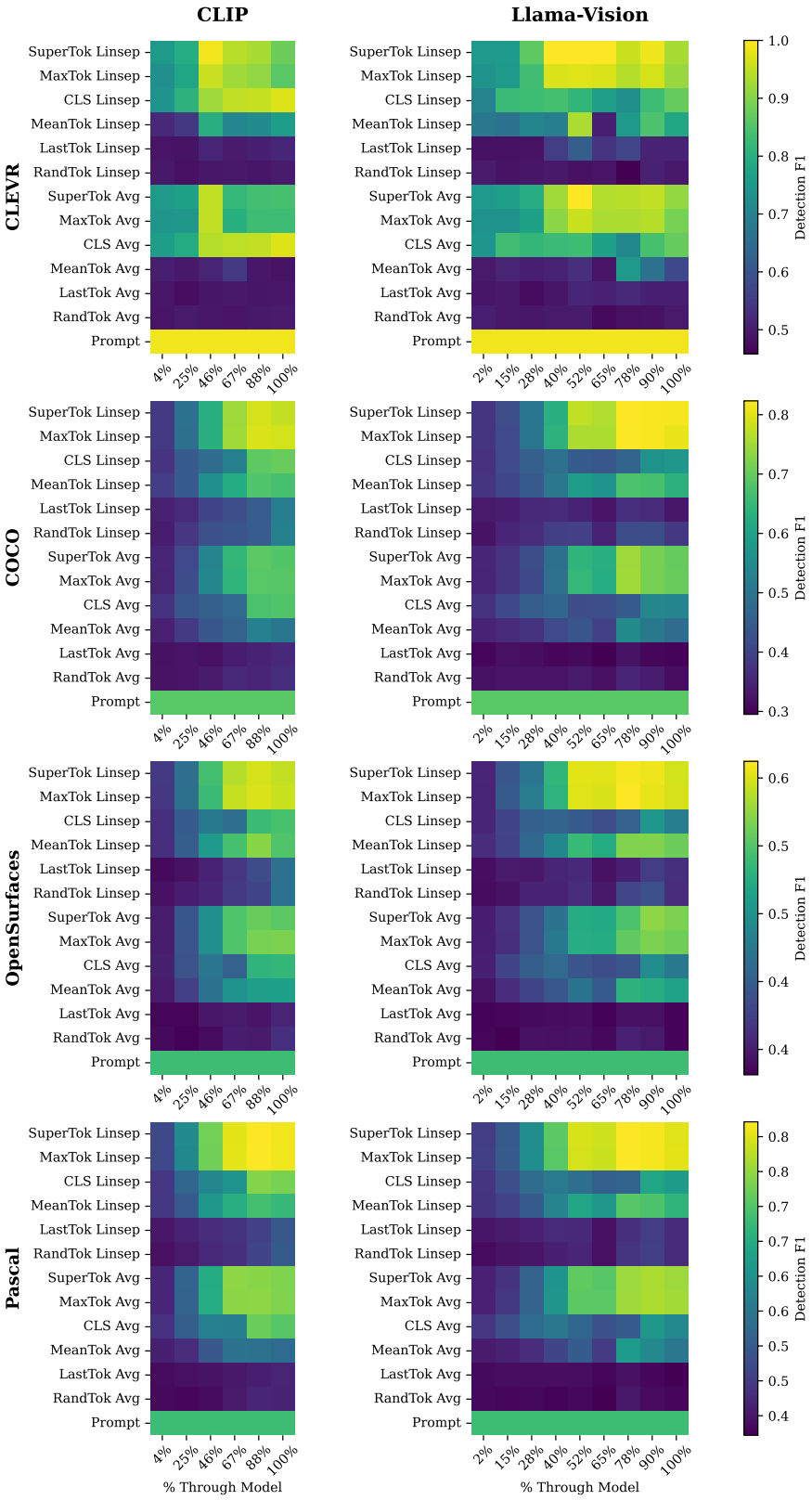


Figure 15: SuperActivator detection across image datasets.

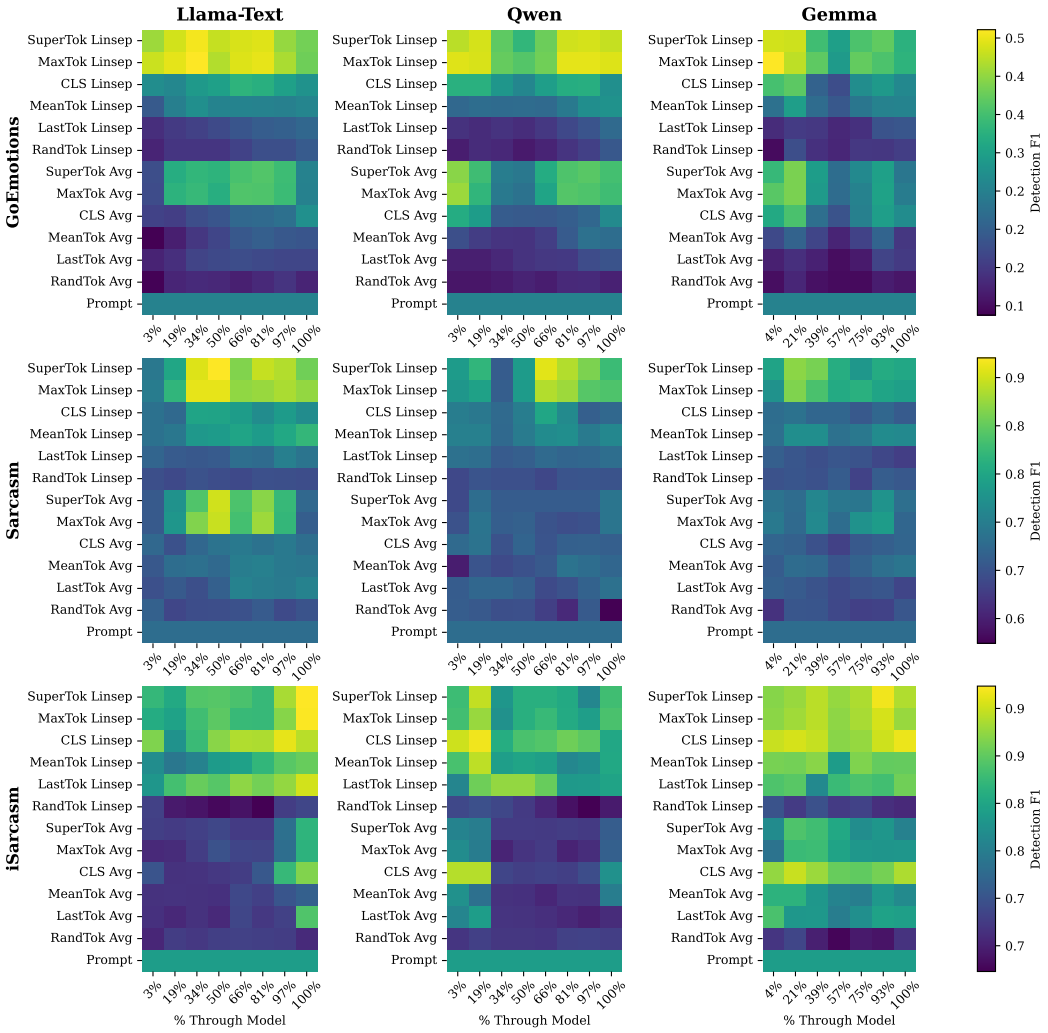


Figure 16: SuperActivator detection across text datasets.

G Ablation: Which Model Layers Yield the Most Separable Concepts?

In this section, we seek to identify where in the model concepts are most separable, that is, at which layers concept vectors achieve their highest detection performance. For each dataset, we plot the frequency of concept vectors that achieve their best F_1 detection scores at each model layer. These trends are shown for the SuperActivator detection scheme as well as for [CLS]-, mean-, and last-token–based detection methods. All results in this analysis use linear separator concept vectors derived from the *LLaMA-3.2-11B-Vision-Instruct* model.

For image datasets with primarily high-level object concepts, such as *COCO* and *Pascal*, the best-performing concept vectors tend to appear in later layers. A similar but less pronounced pattern is observed in *OpenSurfaces*, which contains both high-level objects and lower-level texture concepts. In contrast, *CLEVR*—whose concepts include lower-level properties like color and slightly higher-level ones like shape—shows strong detection performance from both early and late layers, suggesting that different types of concepts emerge at different depths. For the text datasets *Sarcasm*, *iSarcasm*, and *GoEmotions*, a comparable pattern arises: the best-detecting concept vectors most often originate from later layers, though earlier layers also capture meaningful signals for certain concepts.

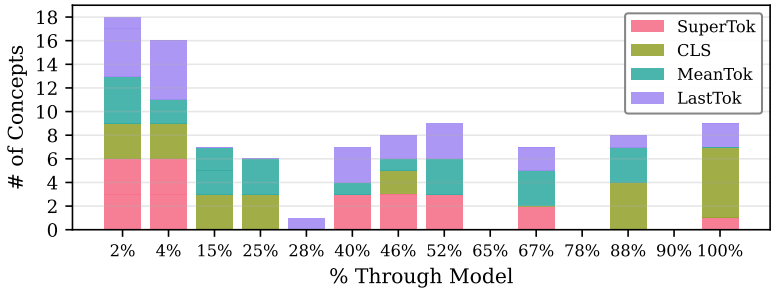


Figure 17: *CLEVR*

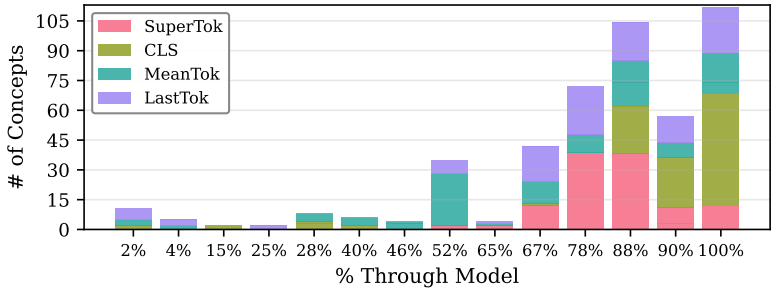


Figure 18: *Coco*

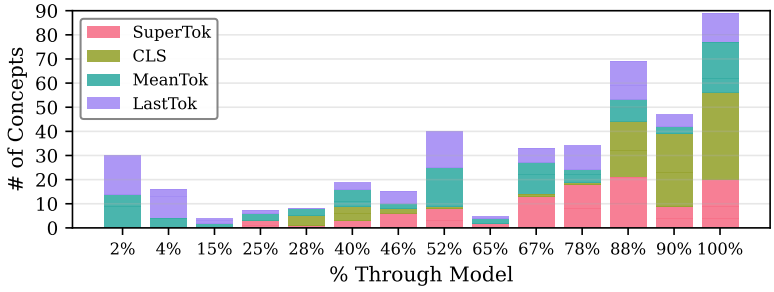


Figure 19: *OpenSurfaces*

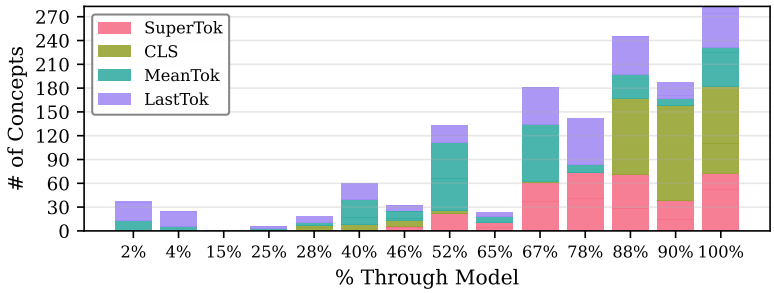


Figure 20: *Pascal*

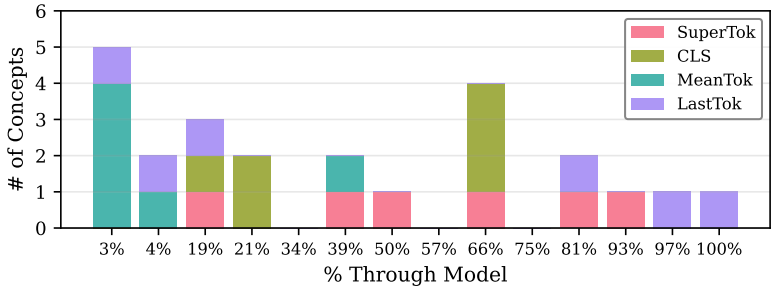


Figure 21: *Sarcasm*

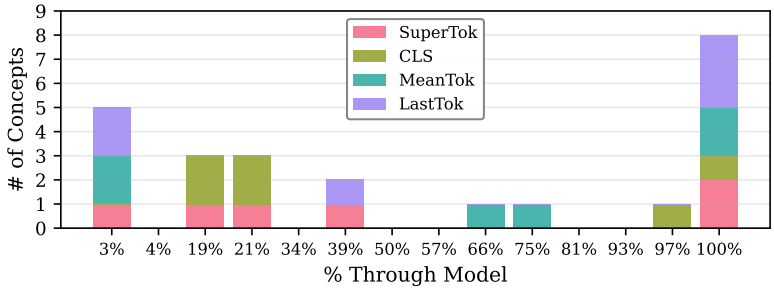


Figure 22: *iSarcasm*

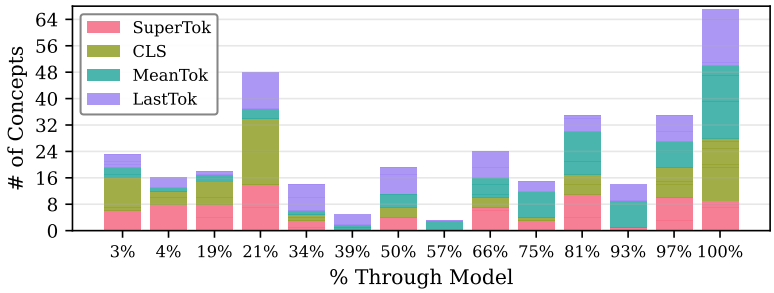


Figure 23: *GoEmotions*

H Ablation: How Does Optimal Sparsity for SuperActivator Detection Vary Across Model Layers?

Next, we analyze how the optimal sparsity levels, δs, for SuperActivator-based concept detection varies across layers in the model. Figures 24 and 25 visualize these results across layers for each model: at every layer, we report the frequency of concepts whose optimal detection occurs at each sparsity level δ, with different colors demarcating the datasets the concepts came from.

Early in the model, the best concept detection via SuperActivators occurs at extremely high sparsity levels ($\delta \approx 0.02$–0.05) for most concepts. However, as shown in Appendix F, these early-layer activations are not yet reliable indicators of concept presence. As we move deeper through the transformer, the best-performing SuperActivators tend to occur at higher δs, meaning that more tokens contribute to concept detection. Even so, the activations remain far from dense, typically involving fewer than half of the true in-concept tokens. Our main takeaway is that the concept signals are expressed most reliably by a small set of activations, no matter the depth that the concepts were extracted from.

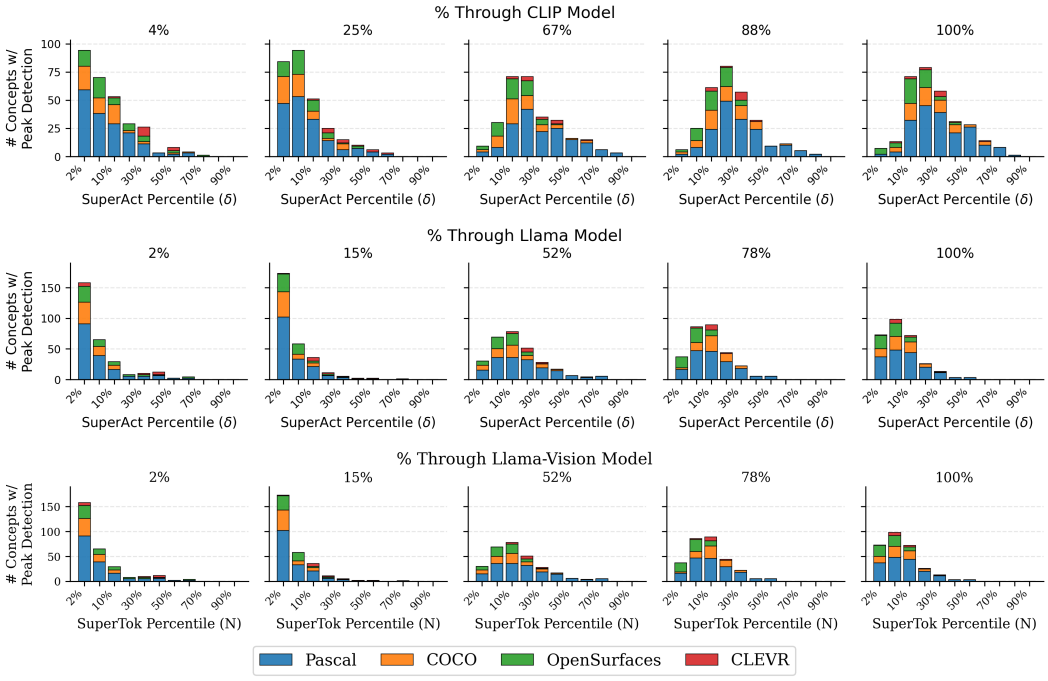


Figure 24: Image Domain – Optimal Sparsity over Layers

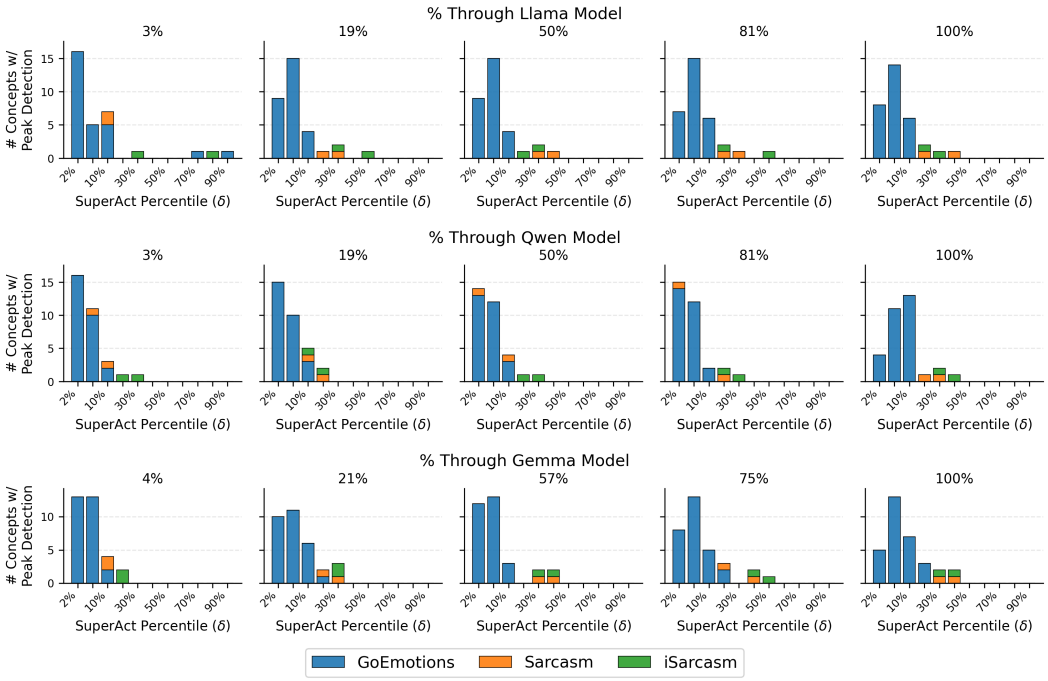


Figure 25: Text Domain – Optimal Sparsity over Layers

I Ablation: How Does Sparsity Affect Average SuperActivator Detection Performance?

In this section, we evaluate SuperActivator-based concept detection performance across varying sparsity levels. The sparsity level δ corresponds to the δ in the SuperActivator definition—thresholds are calibrated using the top δ percent of in-concept token activations. Reported F_1 values represent the average of the per-concept detection F_1, each computed using the corresponding δ, weighted by concept frequency and evaluated at each concept's best-performing layer on the validation set.

Across all model–dataset combinations, we observe that concepts generally achieve their strongest detection performance at low sparsity levels. This supports our broader finding that concept signals are highly concentrated: incorporating additional tokens beyond this sparse subset tends to degrade detection performance.

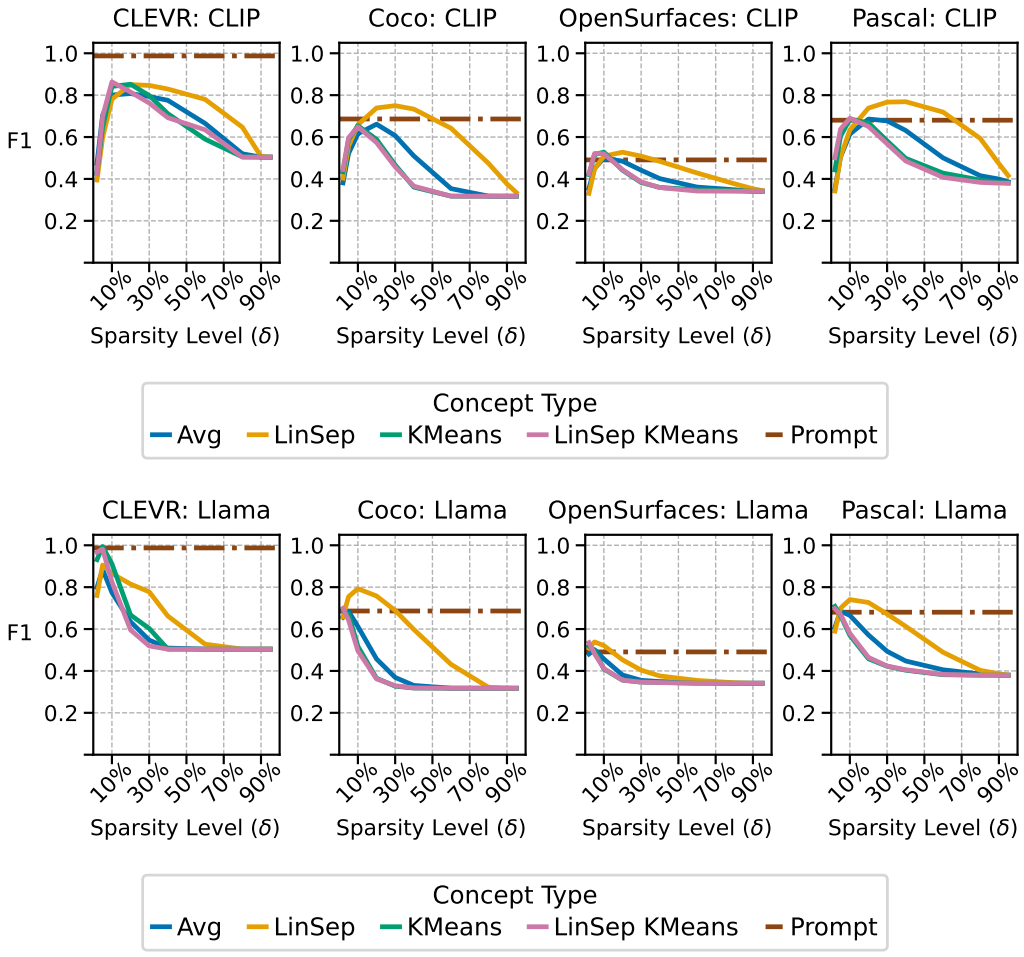


Figure 26: Image Domain – Detection F_1 over Sparsity Level δ

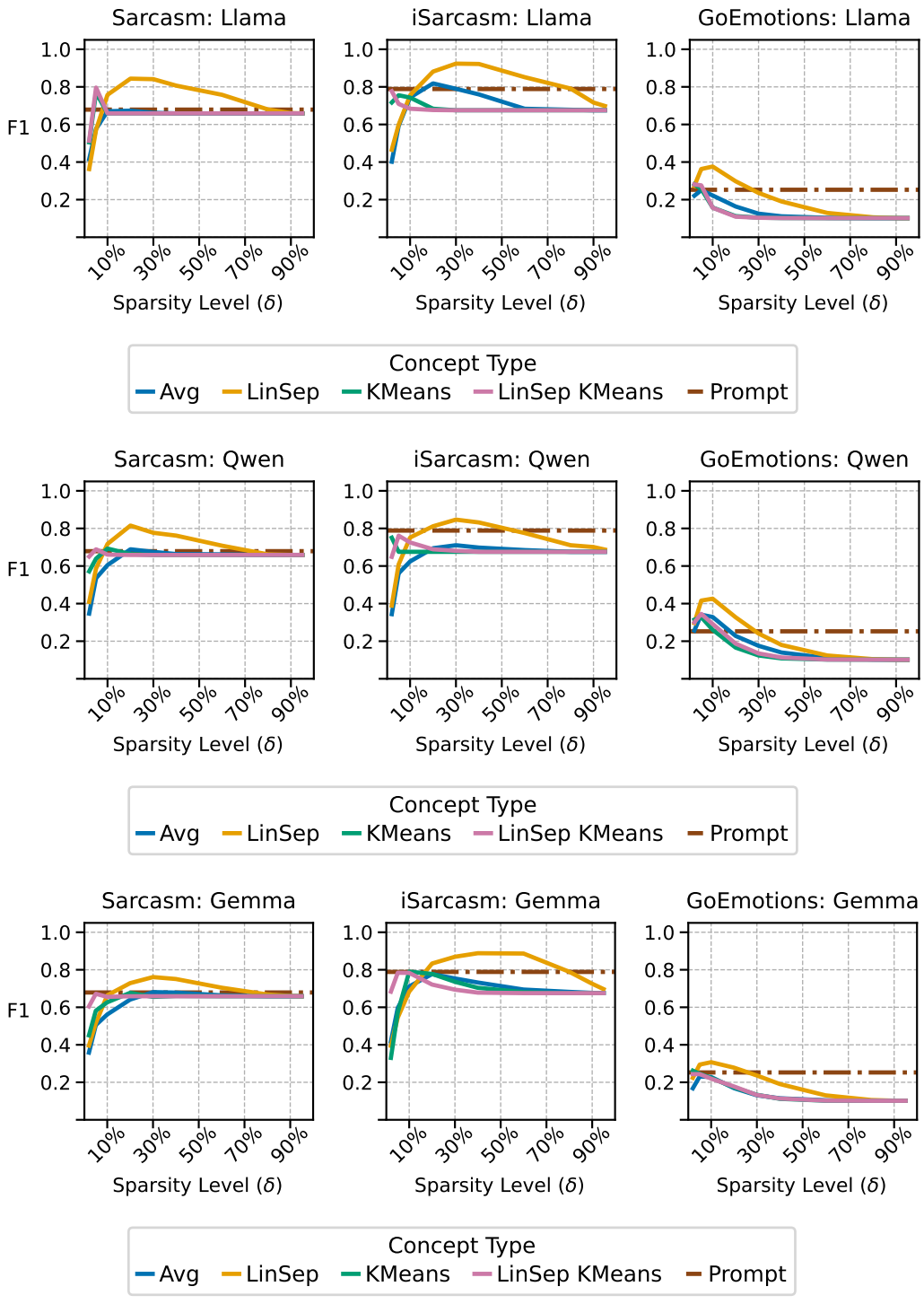


Figure 27: Text Domain – Detection F_1 over Sparsity Level δ

J Qualitative Visualizations of SuperActivators Over Model Layers

Next, we present qualitative examples from each dataset illustrating how the SuperActivator mechanism manifests across layers of the *LLaMA-3.1-11B-Vision-Instruct* model. Each example visualizes linear separator activations for several concepts within a single test sample, along with the corresponding SuperActivators identified using layer-specific, concept-calibrated thresholds.

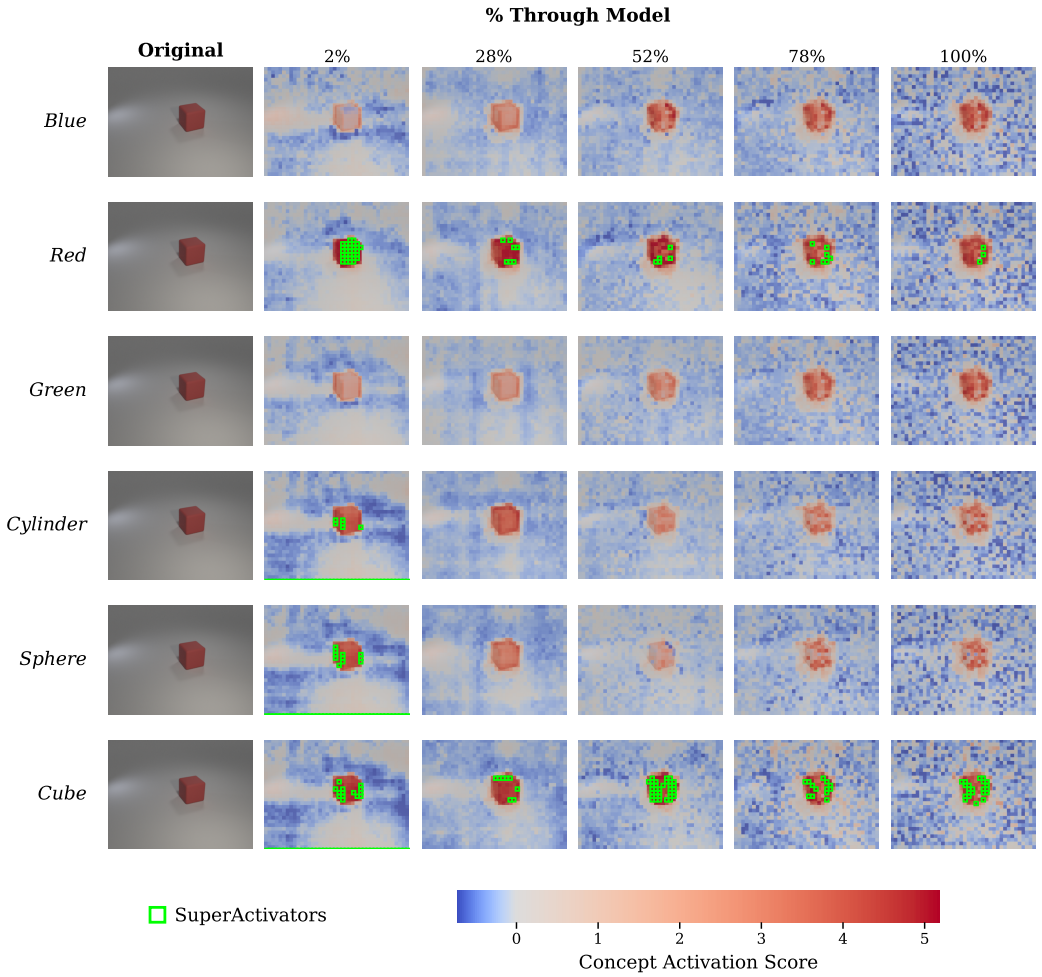


Figure 28: *CLEVR* – SuperActivators Across *LLaMA-3.2-11B-Vision-Instruct* Layers

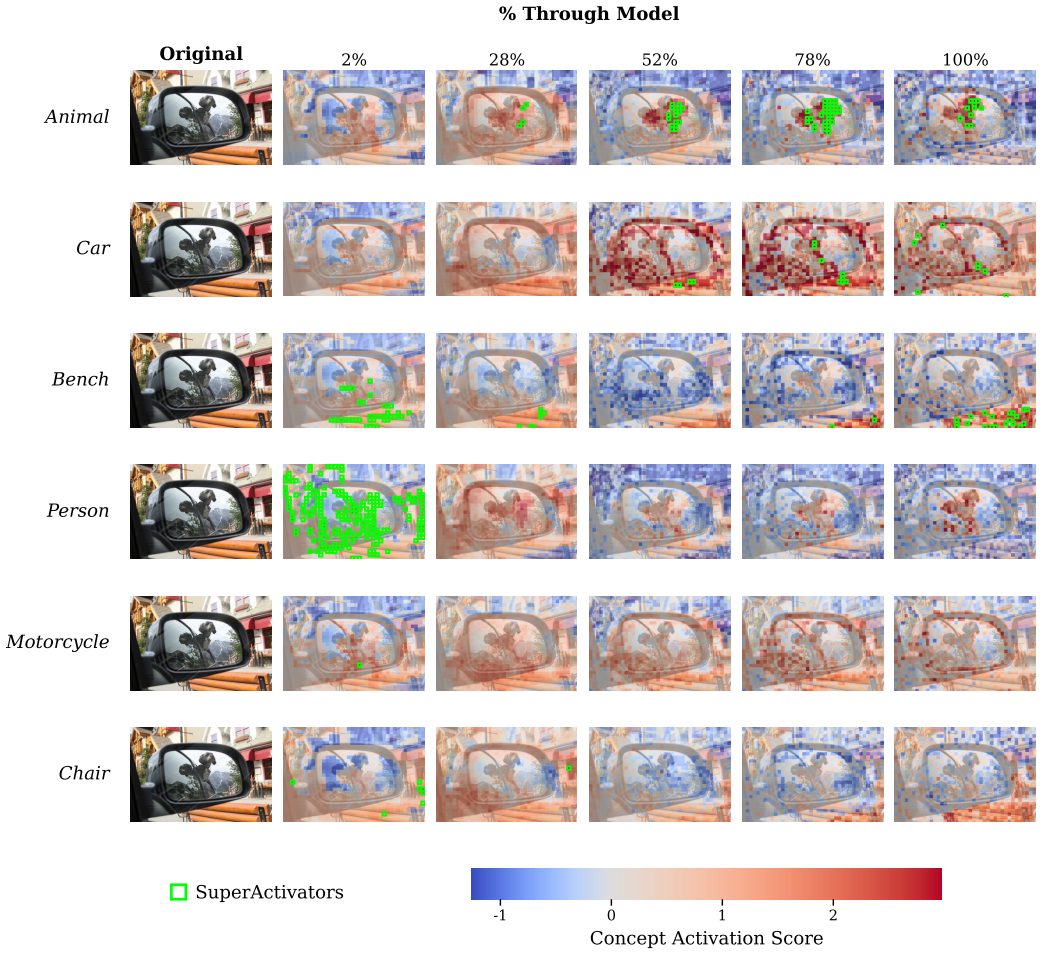


Figure 29: *COCO* – SuperActivators Across *LLaMA-3.2-11B-Vision-Instruct* Layers

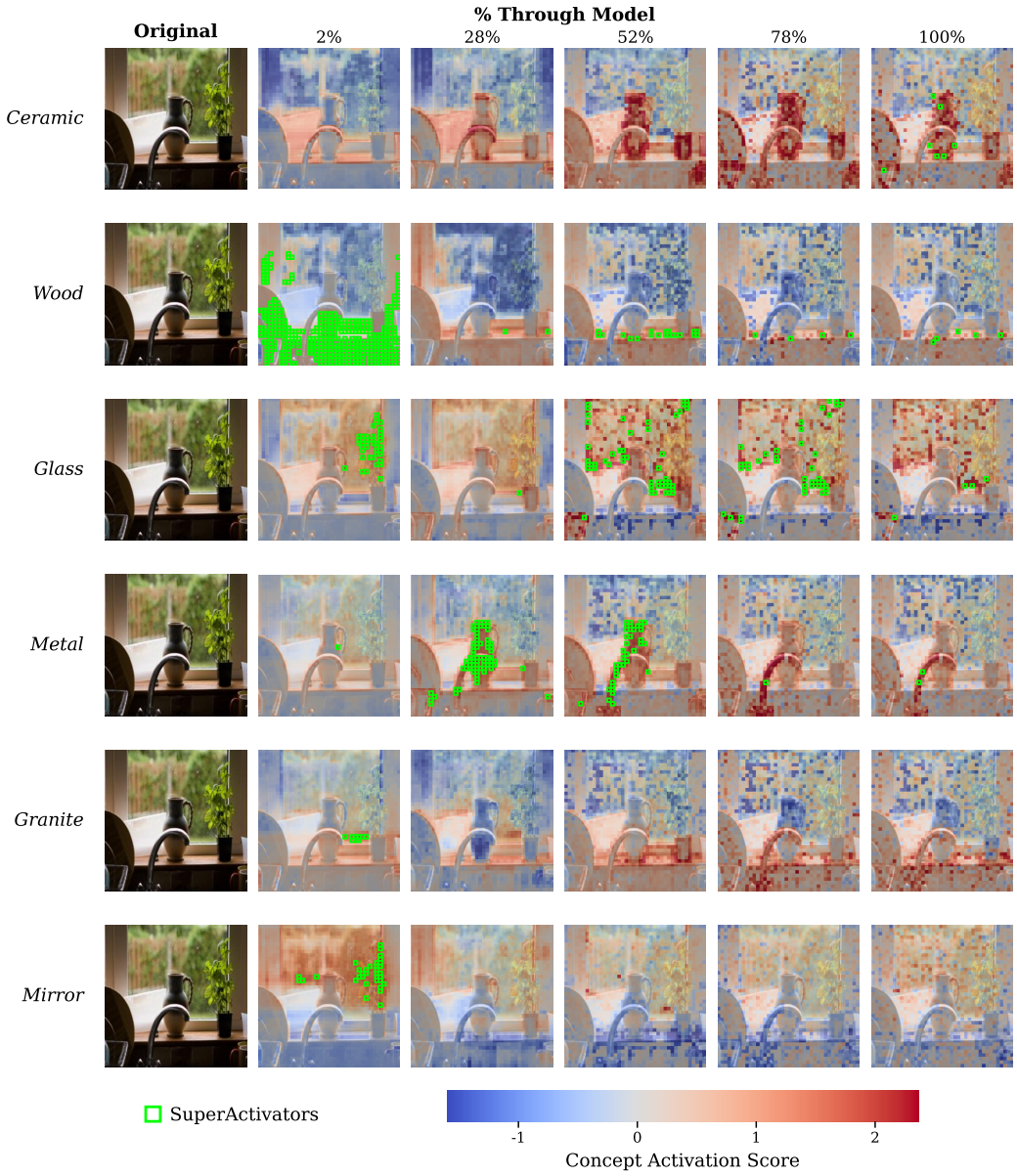


Figure 30: *Broden-OpenSurfaces* – SuperActivators Across *LLaMA-3.2-11B-Vision-Instruct* Layers

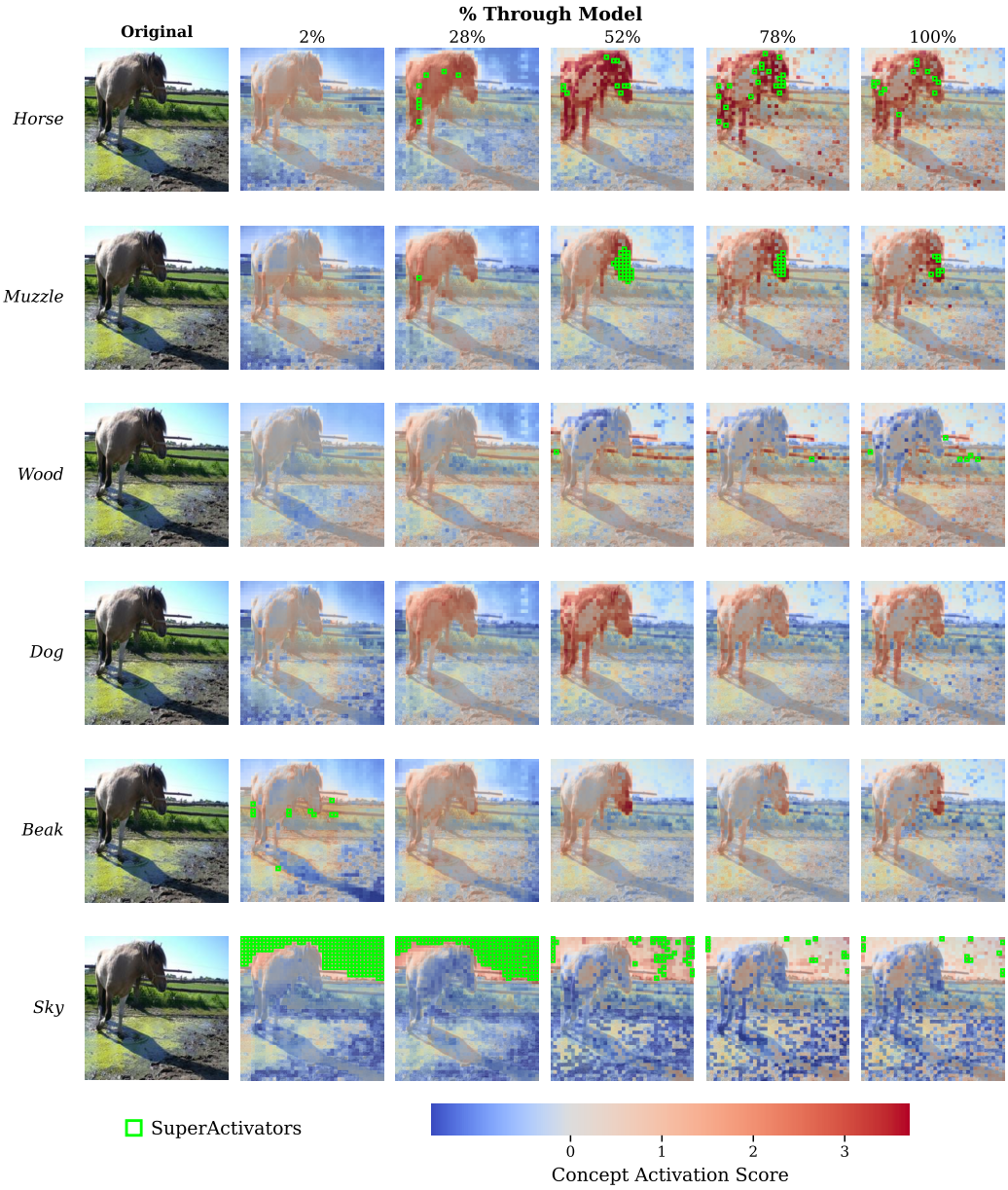


Figure 31: *Broden-Pascal* – SuperActivators Across *LLaMA-3.2-11B-Vision-Instruct* Layers

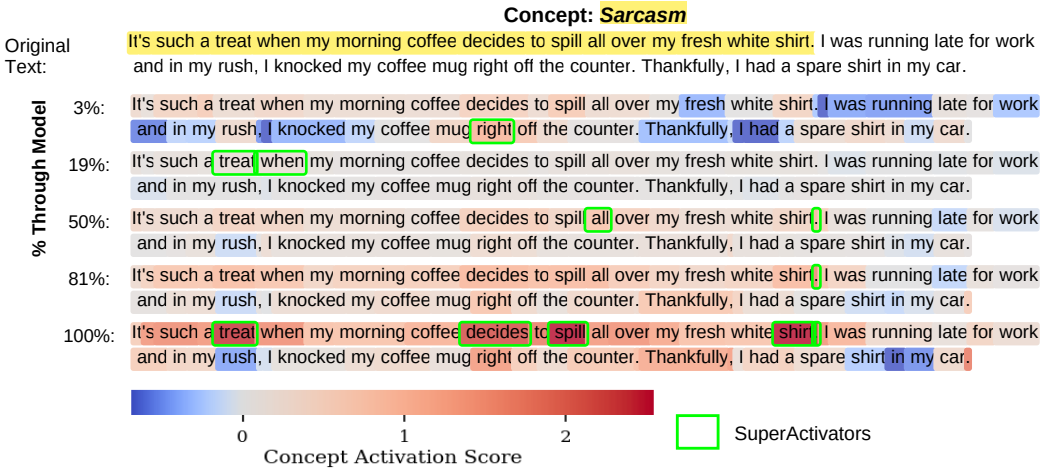


Figure 32: *Sarcasm* – SuperActivators Across *LLaMA-3.2-11B-Vision-Instruct* Layers

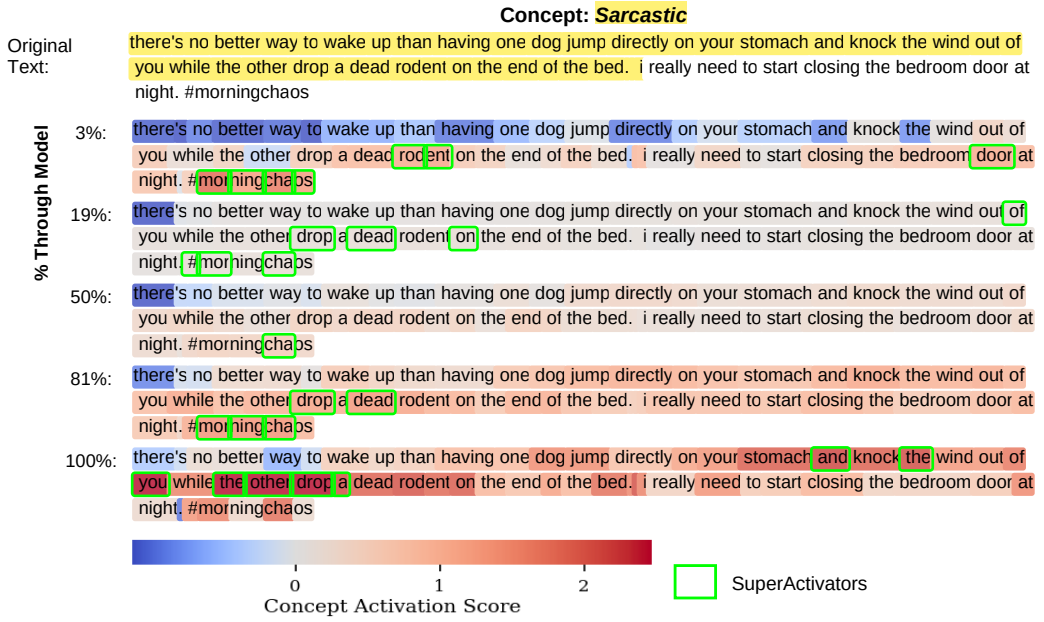


Figure 33: *iSarcasm* – SuperActivators Across *LLaMA-3.2-11B-Vision-Instruct* Layers

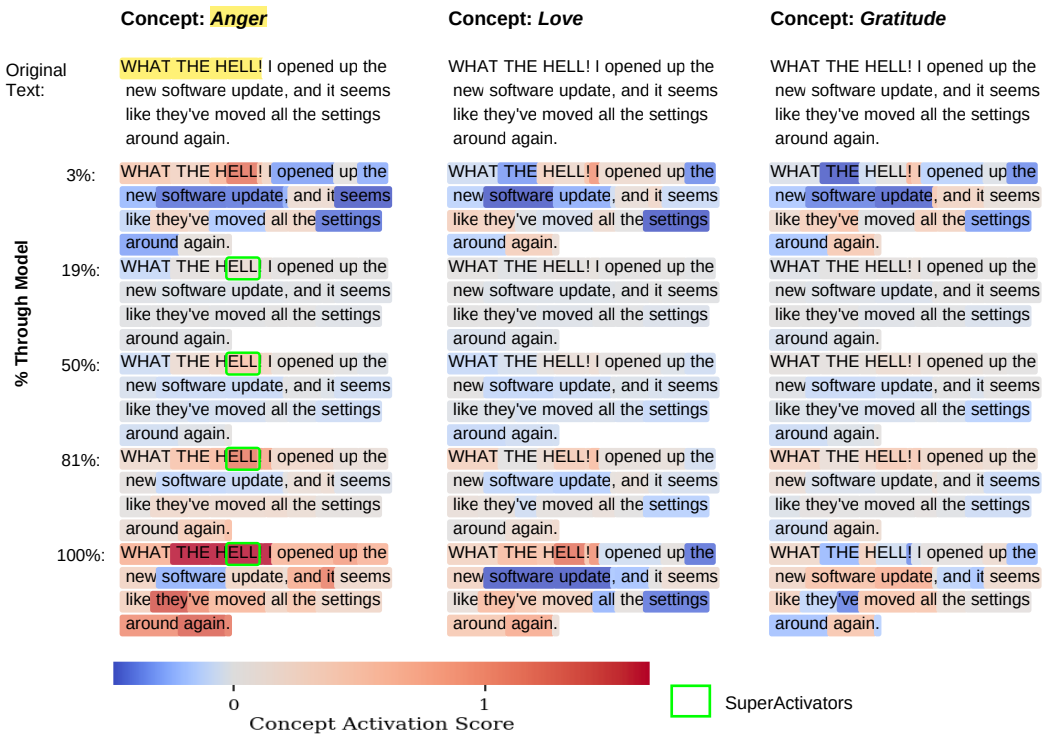


Figure 34: *GoEmotions* – SuperActivators Across *LLaMA-3.2-11B-Vision-Instruct* Layers

K Concept Attribution

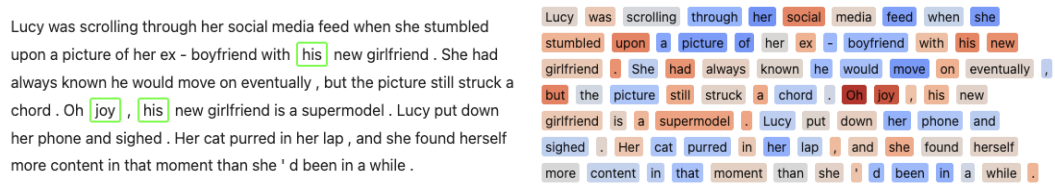
K.1 Attribution Methods

This section provides a brief overview of several attribution methods in which the objective is defined either by a global concept vector v_c or by the average embedding of local SuperActivators.

- **LIME (Local Interpretable Model-agnostic Explanations)** [56] explains an individual prediction by approximating the complex model with a simpler, interpretable model (e.g., a linear model) in the local vicinity of the prediction. It achieves this by generating a new dataset of perturbed samples around the instance being explained and learning the simpler model on this new dataset, weighted by proximity to the original instance.
- **SHAP (SHapley Additive exPlanations)** [57] assigns an importance value to each feature for a particular prediction. Based on cooperative game theory, this value represents the feature's marginal contribution to the model's output, ensuring the sum of all values explains the difference between the model's prediction and a baseline.
- **RISE (Randomized Input Sampling for Explanation)** [58] generates a visual explanation by probing the model with numerous randomly masked versions of an input image. The final importance map is a weighted average of these random masks, where weights are determined by the model's output confidence for each corresponding masked image.
- **SHAP IQ (SHAP Interaction-aware exPlanations for Quantifying feature importance)** [59] extends the SHAP framework to quantify the effects of feature interactions. Beyond calculating the main effect of each feature, it also computes interaction indices to provide a more complete picture of how combinations of features jointly influence a prediction.
- **IntGrad (Integrated Gradients)** [60] calculates the importance of each input feature by integrating the gradients of the model's output with respect to the feature's inputs. This integration is performed along a straight-line path from a baseline input (e.g., a black image) to the actual input, satisfying key axioms like sensitivity.
- **Grad-CAM (Gradient-weighted Class Activation Mapping)** [61] produces a coarse localization map for CNNs by using the gradients of the target class score with respect to the feature maps of the final convolutional layer. These gradients are used to compute a weighted combination of the activation maps, highlighting important image regions.
- **FullGrad** [62] enhances gradient-based explanations by aggregating gradient information from all layers of a neural network. It combines the input gradients with bias gradients from all intermediate feature maps to capture more comprehensive feature representations, resulting in more detailed saliency maps.
- **CALM (Class Activation Latent Mapping)** [63] improves on Class Activation Mapping (CAM) by introducing a probabilistic latent variable that directly represents the location of the most important visual cue for a model's prediction. Trained with the Expectation-Maximization (EM) algorithm, the method outputs a probability map showing the likelihood that each pixel is the critical cue for the decision.
- **MFABA (More Faithful and Accelerated Boundary-based Attribution)** [86] is a boundary-based attribution method that constructs a path from an input toward the decision boundary. Along this path, it uses a second-order Taylor expansion of the loss function to better approximate how the model's output or loss changes. The resulting attribution scores reflect how much each feature contributes to pushing the input toward or away from the boundary.

K.2 Additional Results for Concept Attribution

This section presents the full results for concept attribution across all experimental configurations, which were summarized in Table 2 in the main text. These detailed tables are provided to demonstrate that our main findings are consistent across all individual concepts and experimental settings. As these results confirm, using the average embedding of local SuperActivators as the explanation objective consistently leads to better performance than using the concept vector directly.



(a) Original text (*Sarcasm* SuperActivators label in green)

(b) *Sarcasm* global concept vector attribution map

Figure 35: **SuperActivators produce attribution maps that better match the true sarcastic cues.** Shown are token-level attributions for the concept *Sarcasm* on a sample from the Sarcasm dataset, using LLaMA token-level linear-separator concepts with LIME-based attribution. Red indicates high alignment and blue indicates low alignment. In (b), many highly aligned tokens fall outside the labeled sarcastic region, while SuperActivators (green boxes) align more closely with the ground-truth cues.

We present our results across seven tables, each corresponding to one dataset and jointly evaluating both supervised and unsupervised concept representations. For every dataset—four image tasks (Tables 3, 4, 5, and 6) and three text tasks (Tables 7, 8, and 9)—we report the average F_1 score across all concepts, weighted by their frequency in the test set (Appendix C.5). Each table compares two concept extraction methods (Clustering vs. Linear Separator) and two attribution objectives (global concept vector vs. average local SuperActivators patch embedding). Within each table:

- **Avg** denotes supervised clustering-based concepts
- **Linsep** denotes supervised linear-separator concepts
- **K-Means** denotes unsupervised clustering-based concepts
- **K-Linsep** denotes unsupervised linear-separator concepts

Additional details on concept extraction and evaluation procedures are provided in Appendix C.2.

K.3 Qualitative Example Showing SuperActivators for Improved Concept Attribution

Figure 6 further illustrates the advantage: attribution using SuperActivators for the concept *person* provides better coverage for the full target object while avoiding irrelevant regions such as tables, which the global vector incorrectly highlights.

Figure 35 shows a similar pattern on text: SuperActivators attribution for *sarcasm* focuses on the true sarcastic cues while avoiding irrelevant tokens that the global concept vector incorrectly highlights.

Table 3: Average F1 for the CLEVR Dataset.

Attribution Method	Concept Type	CLIP		Llama	
		Concept	SuperActivators	Concept	SuperActivators
CosSim	Avg	**0.60 ± 0.02**	**0.60 ± 0.01**	**0.78 ± 0.01**	0.55 ± 0.03
	Linsep	**0.65 ± 0.01**	0.61 ± 0.03	**0.85 ± 0.02**	0.54 ± 0.01
	K-Means	0.63 ± 0.02	**0.64 ± 0.01**	**0.46 ± 0.01**	0.43 ± 0.03
	K-Linsep	**0.60 ± 0.01**	0.59 ± 0.03	**0.38 ± 0.02**	0.33 ± 0.01
LIME	Avg	0.49 ± 0.02	**0.55 ± 0.04**	0.76 ± 0.03	**0.81 ± 0.02**
	Linsep	0.49 ± 0.00	**0.68 ± 0.01**	0.70 ± 0.01	**0.85 ± 0.01**
	K-Means	0.52 ± 0.03	**0.61 ± 0.01**	0.76 ± 0.01	**0.81 ± 0.02**
	K-Linsep	0.52 ± 0.02	**0.77 ± 0.03**	0.68 ± 0.03	**0.83 ± 0.01**
SHAP	Avg	0.51 ± 0.01	**0.53 ± 0.02**	0.75 ± 0.02	**0.80 ± 0.03**
	Linsep	0.52 ± 0.03	**0.58 ± 0.01**	0.75 ± 0.01	**0.80 ± 0.01**
	K-Means	0.51 ± 0.01	**0.53 ± 0.02**	0.75 ± 0.02	**0.80 ± 0.01**
	K-Linsep	0.52 ± 0.03	**0.58 ± 0.01**	0.75 ± 0.01	**0.80 ± 0.03**
RISE	Avg	**0.53 ± 0.02**	**0.53 ± 0.03**	0.55 ± 0.03	**0.56 ± 0.02**
	Linsep	0.58 ± 0.01	**0.59 ± 0.02**	0.60 ± 0.02	**0.63 ± 0.01**
	K-Means	**0.53 ± 0.02**	**0.53 ± 0.01**	0.55 ± 0.03	**0.56 ± 0.02**
	K-Linsep	0.58 ± 0.01	**0.59 ± 0.03**	0.60 ± 0.01	**0.63 ± 0.02**
SHAP IQ	Avg	0.52 ± 0.04	**0.53 ± 0.01**	0.55 ± 0.01	**0.58 ± 0.02**
	Linsep	**0.58 ± 0.02**	**0.58 ± 0.03**	0.60 ± 0.03	**0.61 ± 0.01**
	K-Means	0.52 ± 0.03	**0.53 ± 0.02**	0.55 ± 0.02	**0.58 ± 0.01**
	K-Linsep	**0.58 ± 0.01**	**0.58 ± 0.02**	0.60 ± 0.01	**0.61 ± 0.03**
IntGrad	Avg	0.46 ± 0.01	**0.53 ± 0.03**	0.77 ± 0.02	**0.80 ± 0.02**
	Linsep	0.49 ± 0.03	**0.55 ± 0.01**	0.72 ± 0.01	**0.78 ± 0.03**
	K-Means	**0.47 ± 0.02**	**0.47 ± 0.01**	0.56 ± 0.03	**0.58 ± 0.02**
	K-Linsep	0.58 ± 0.01	**0.59 ± 0.03**	0.62 ± 0.01	**0.64 ± 0.02**
GradCAM	Avg	0.45 ± 0.02	**0.48 ± 0.01**	0.50 ± 0.03	**0.52 ± 0.01**
	Linsep	**0.48 ± 0.01**	**0.48 ± 0.02**	0.50 ± 0.02	**0.52 ± 0.02**
	K-Means	0.41 ± 0.03	**0.45 ± 0.02**	**0.50 ± 0.02**	0.47 ± 0.01
	K-Linsep	**0.48 ± 0.01**	0.46 ± 0.02	**0.48 ± 0.01**	0.49 ± 0.03
FullGrad	Avg	**0.46 ± 0.02**	**0.46 ± 0.03**	**0.47 ± 0.01**	0.49 ± 0.02
	Linsep	0.50 ± 0.01	**0.52 ± 0.02**	0.51 ± 0.02	**0.55 ± 0.01**
	K-Means	**0.45 ± 0.02**	0.42 ± 0.01	**0.42 ± 0.03**	0.45 ± 0.02
	K-Linsep	**0.49 ± 0.01**	**0.49 ± 0.03**	0.50 ± 0.01	**0.53 ± 0.02**
CALM	Avg	0.48 ± 0.03	**0.52 ± 0.01**	0.49 ± 0.03	**0.53 ± 0.02**
	Linsep	0.55 ± 0.02	**0.56 ± 0.02**	**0.57 ± 0.01**	**0.57 ± 0.03**
	K-Means	0.44 ± 0.03	**0.50 ± 0.02**	0.46 ± 0.02	**0.48 ± 0.01**
	K-Linsep	0.50 ± 0.01	**0.54 ± 0.02**	0.53 ± 0.01	**0.54 ± 0.03**
MFABA	Avg	0.50 ± 0.01	**0.51 ± 0.01**	0.51 ± 0.02	**0.53 ± 0.01**
	Linsep	**0.55 ± 0.03**	**0.55 ± 0.02**	0.56 ± 0.01	**0.58 ± 0.03**
	K-Means	0.45 ± 0.02	**0.48 ± 0.01**	0.47 ± 0.03	**0.52 ± 0.02**
	K-Linsep	**0.51 ± 0.01**	0.50 ± 0.03	0.54 ± 0.01	**0.55 ± 0.02**

Table 4: Average F1 for the COCO Dataset.

Attribution Method	Concept Type	CLIP		Llama	
		Concept	SuperActivators	Concept	SuperActivators
CosSim	Avg	**0.43 ± 0.03**	0.40 ± 0.02	0.36 ± 0.02	**0.37 ± 0.01**
	Linsep	**0.52 ± 0.02**	0.45 ± 0.00	**0.46 ± 0.03**	0.44 ± 0.02
	K-Means	0.34 ± 0.03	**0.37 ± 0.02**	0.22 ± 0.02	**0.28 ± 0.01**
	K-Linsep	0.33 ± 0.02	**0.36 ± 0.01**	0.23 ± 0.03	**0.26 ± 0.02**
LIME	Avg	0.32 ± 0.01	**0.38 ± 0.02**	0.47 ± 0.01	**0.51 ± 0.02**
	Linsep	0.29 ± 0.02	**0.40 ± 0.03**	0.49 ± 0.02	**0.50 ± 0.03**
	K-Means	0.36 ± 0.02	**0.38 ± 0.03**	0.45 ± 0.03	**0.52 ± 0.01**
	K-Linsep	0.38 ± 0.01	**0.41 ± 0.02**	0.49 ± 0.02	**0.55 ± 0.03**
SHAP	Avg	0.34 ± 0.03	**0.38 ± 0.01**	0.48 ± 0.03	**0.51 ± 0.01**
	Linsep	0.35 ± 0.01	**0.37 ± 0.02**	0.49 ± 0.02	**0.55 ± 0.04**
	K-Means	0.34 ± 0.03	**0.38 ± 0.01**	0.48 ± 0.03	**0.51 ± 0.01**
	K-Linsep	0.35 ± 0.02	**0.37 ± 0.03**	0.49 ± 0.02	**0.53 ± 0.01**
RISE	Avg	**0.34 ± 0.01**	**0.34 ± 0.02**	0.36 ± 0.01	**0.38 ± 0.01**
	Linsep	0.35 ± 0.02	**0.38 ± 0.03**	0.35 ± 0.03	**0.40 ± 0.02**
	K-Means	**0.34 ± 0.03**	**0.34 ± 0.02**	0.36 ± 0.01	**0.38 ± 0.03**
	K-Linsep	0.35 ± 0.02	**0.38 ± 0.01**	0.35 ± 0.03	**0.40 ± 0.02**
SHAP IQ	Avg	0.33 ± 0.03	**0.35 ± 0.02**	0.35 ± 0.02	**0.36 ± 0.01**
	Linsep	0.34 ± 0.01	**0.37 ± 0.01**	0.36 ± 0.01	**0.38 ± 0.03**
	K-Means	0.33 ± 0.01	**0.35 ± 0.03**	0.35 ± 0.02	**0.36 ± 0.01**
	K-Linsep	0.34 ± 0.03	**0.37 ± 0.01**	0.36 ± 0.02	**0.38 ± 0.01**
IntGrad	Avg	0.30 ± 0.02	**0.33 ± 0.02**	0.42 ± 0.03	**0.45 ± 0.01**
	Linsep	0.28 ± 0.00	**0.35 ± 0.04**	0.43 ± 0.02	**0.48 ± 0.01**
	K-Means	0.28 ± 0.03	**0.31 ± 0.02**	**0.48 ± 0.01**	0.47 ± 0.03
	K-Linsep	0.31 ± 0.02	**0.35 ± 0.01**	0.38 ± 0.03	**0.39 ± 0.01**
GradCAM	Avg	**0.31 ± 0.03**	**0.31 ± 0.01**	0.32 ± 0.02	**0.35 ± 0.03**
	Linsep	0.37 ± 0.01	**0.38 ± 0.02**	**0.37 ± 0.01**	**0.37 ± 0.02**
	K-Means	0.28 ± 0.01	**0.31 ± 0.03**	0.31 ± 0.03	**0.33 ± 0.02**
	K-Linsep	0.35 ± 0.03	**0.36 ± 0.01**	0.36 ± 0.02	**0.34 ± 0.01**
FullGrad	Avg	**0.33 ± 0.02**	0.32 ± 0.01	0.35 ± 0.03	**0.38 ± 0.01**
	Linsep	**0.43 ± 0.01**	**0.43 ± 0.00**	**0.39 ± 0.01**	**0.39 ± 0.03**
	K-Means	0.29 ± 0.03	**0.31 ± 0.02**	0.30 ± 0.01	**0.33 ± 0.03**
	K-Linsep	0.35 ± 0.02	**0.39 ± 0.01**	0.37 ± 0.03	**0.34 ± 0.01**
CALM	Avg	**0.32 ± 0.02**	**0.32 ± 0.03**	0.30 ± 0.01	**0.29 ± 0.02**
	Linsep	**0.42 ± 0.01**	**0.42 ± 0.01**	0.38 ± 0.02	**0.41 ± 0.01**
	K-Means	**0.29 ± 0.01**	**0.29 ± 0.03**	0.26 ± 0.02	**0.25 ± 0.02**
	K-Linsep	0.35 ± 0.02	**0.39 ± 0.01**	0.35 ± 0.02	**0.36 ± 0.01**
MFABA	Avg	0.31 ± 0.04	**0.37 ± 0.02**	0.33 ± 0.03	**0.34 ± 0.02**
	Linsep	0.33 ± 0.01	**0.39 ± 0.03**	0.35 ± 0.02	**0.39 ± 0.01**
	K-Means	0.29 ± 0.03	**0.33 ± 0.02**	0.28 ± 0.01	**0.32 ± 0.03**
	K-Linsep	0.30 ± 0.02	**0.35 ± 0.01**	0.33 ± 0.03	**0.36 ± 0.01**

Table 5: Average F1 for the OpenSurfaces Dataset.

Attribution Method	Concept Type	CLIP		Llama	
		Concept	SuperActivators	Concept	SuperActivators
CosSim	Avg	**0.22 ± 0.01**	0.18 ± 0.04	**0.19 ± 0.03**	0.15 ± 0.02
	Linsep	**0.28 ± 0.03**	0.22 ± 0.02	**0.23 ± 0.01**	0.17 ± 0.01
	K-Means	**0.19 ± 0.01**	**0.19 ± 0.03**	0.14 ± 0.03	**0.15 ± 0.02**
	K-Linsep	**0.19 ± 0.03**	0.18 ± 0.02	**0.15 ± 0.01**	0.14 ± 0.03
LIME	Avg	0.42 ± 0.03	**0.50 ± 0.01**	0.55 ± 0.03	**0.62 ± 0.01**
	Linsep	0.46 ± 0.01	**0.50 ± 0.03**	0.60 ± 0.01	**0.68 ± 0.02**
	K-Means	0.37 ± 0.01	**0.41 ± 0.02**	**0.37 ± 0.02**	**0.37 ± 0.03**
	K-Linsep	0.39 ± 0.03	**0.41 ± 0.01**	0.38 ± 0.01	**0.39 ± 0.02**
SHAP	Avg	0.40 ± 0.02	**0.42 ± 0.04**	0.53 ± 0.02	**0.57 ± 0.03**
	Linsep	0.42 ± 0.02	**0.44 ± 0.01**	0.55 ± 0.03	**0.56 ± 0.01**
	K-Means	0.40 ± 0.02	**0.42 ± 0.03**	0.53 ± 0.02	**0.57 ± 0.03**
	K-Linsep	0.42 ± 0.01	**0.44 ± 0.02**	0.55 ± 0.03	**0.56 ± 0.01**
RISE	Avg	0.40 ± 0.04	**0.42 ± 0.01**	0.51 ± 0.02	**0.52 ± 0.03**
	Linsep	0.43 ± 0.01	**0.45 ± 0.02**	0.53 ± 0.01	**0.55 ± 0.02**
	K-Means	0.40 ± 0.01	**0.42 ± 0.03**	0.51 ± 0.02	**0.52 ± 0.01**
	K-Linsep	0.43 ± 0.03	**0.45 ± 0.02**	0.53 ± 0.01	**0.55 ± 0.02**
SHAP IQ	Avg	0.40 ± 0.02	**0.43 ± 0.01**	0.51 ± 0.03	**0.53 ± 0.02**
	Linsep	0.42 ± 0.03	**0.45 ± 0.02**	**0.52 ± 0.01**	**0.52 ± 0.02**
	K-Means	0.40 ± 0.02	**0.43 ± 0.01**	0.51 ± 0.03	**0.53 ± 0.02**
	K-Linsep	0.42 ± 0.02	**0.45 ± 0.03**	**0.52 ± 0.01**	**0.52 ± 0.02**
IntGrad	Avg	0.43 ± 0.01	**0.51 ± 0.02**	0.46 ± 0.02	**0.47 ± 0.03**
	Linsep	0.44 ± 0.02	**0.49 ± 0.02**	0.56 ± 0.01	**0.62 ± 0.02**
	K-Means	0.33 ± 0.01	**0.34 ± 0.03**	0.32 ± 0.02	**0.35 ± 0.01**
	K-Linsep	**0.35 ± 0.03**	**0.35 ± 0.02**	0.34 ± 0.02	**0.35 ± 0.03**
GradCAM	Avg	0.41 ± 0.02	**0.43 ± 0.03**	0.45 ± 0.01	**0.46 ± 0.02**
	Linsep	0.44 ± 0.01	**0.46 ± 0.01**	0.45 ± 0.03	**0.51 ± 0.01**
	K-Means	0.36 ± 0.02	**0.40 ± 0.01**	**0.43 ± 0.01**	0.42 ± 0.03
	K-Linsep	0.42 ± 0.02	**0.43 ± 0.03**	0.44 ± 0.01	**0.46 ± 0.02**
FullGrad	Avg	0.38 ± 0.03	**0.41 ± 0.02**	0.40 ± 0.02	**0.41 ± 0.01**
	Linsep	0.42 ± 0.04	**0.45 ± 0.01**	0.43 ± 0.01	**0.47 ± 0.02**
	K-Means	0.36 ± 0.01	**0.37 ± 0.03**	0.36 ± 0.02	**0.38 ± 0.01**
	K-Linsep	0.38 ± 0.03	**0.40 ± 0.02**	0.41 ± 0.01	**0.44 ± 0.02**
CALM	Avg	0.33 ± 0.01	**0.35 ± 0.01**	0.35 ± 0.02	**0.37 ± 0.01**
	Linsep	0.35 ± 0.02	**0.38 ± 0.03**	0.36 ± 0.01	**0.41 ± 0.03**
	K-Means	0.29 ± 0.02	**0.32 ± 0.01**	0.33 ± 0.01	**0.36 ± 0.03**
	K-Linsep	0.32 ± 0.02	**0.34 ± 0.03**	0.34 ± 0.02	**0.39 ± 0.01**
MFABA	Avg	0.42 ± 0.02	**0.44 ± 0.03**	**0.44 ± 0.01**	**0.44 ± 0.02**
	Linsep	0.45 ± 0.01	**0.48 ± 0.01**	0.44 ± 0.02	**0.47 ± 0.03**
	K-Means	**0.40 ± 0.01**	**0.40 ± 0.03**	0.42 ± 0.01	**0.41 ± 0.02**
	K-Linsep	0.43 ± 0.03	**0.45 ± 0.02**	0.42 ± 0.03	**0.44 ± 0.01**

Table 6: Average F1 for the Pascal Dataset.

Attribution Method	Concept Type	CLIP		Llama	
		Concept	SuperActivators	Concept	SuperActivators
CosSim	Avg	**0.42 ± 0.02**	0.35 ± 0.01	**0.40 ± 0.01**	0.29 ± 0.04
	Linsep	**0.54 ± 0.01**	0.42 ± 0.03	**0.46 ± 0.02**	0.33 ± 0.03
	K-Means	0.27 ± 0.02	**0.33 ± 0.01**	0.22 ± 0.01	**0.24 ± 0.03**
	K-Linsep	0.24 ± 0.01	**0.30 ± 0.03**	0.22 ± 0.02	**0.24 ± 0.01**
LIME	Avg	0.50 ± 0.02	**0.52 ± 0.02**	0.69 ± 0.02	**0.71 ± 0.03**
	Linsep	0.51 ± 0.03	**0.55 ± 0.01**	0.71 ± 0.03	**0.72 ± 0.01**
	K-Means	0.33 ± 0.03	**0.34 ± 0.01**	0.33 ± 0.01	**0.32 ± 0.02**
	K-Linsep	0.36 ± 0.02	**0.35 ± 0.03**	**0.33 ± 0.03**	**0.33 ± 0.01**
SHAP	Avg	0.48 ± 0.01	**0.52 ± 0.03**	0.65 ± 0.01	**0.70 ± 0.02**
	Linsep	0.50 ± 0.00	**0.52 ± 0.02**	0.69 ± 0.02	**0.72 ± 0.01**
	K-Means	0.48 ± 0.01	**0.52 ± 0.02**	0.65 ± 0.02	**0.70 ± 0.01**
	K-Linsep	0.50 ± 0.03	**0.52 ± 0.01**	0.69 ± 0.01	**0.72 ± 0.03**
RISE	Avg	0.50 ± 0.03	**0.51 ± 0.01**	0.52 ± 0.01	**0.55 ± 0.03**
	Linsep	**0.54 ± 0.03**	**0.54 ± 0.02**	0.55 ± 0.02	**0.58 ± 0.01**
	K-Means	0.50 ± 0.02	**0.51 ± 0.01**	0.52 ± 0.01	**0.55 ± 0.03**
	K-Linsep	**0.54 ± 0.01**	**0.54 ± 0.03**	0.55 ± 0.02	**0.58 ± 0.01**
SHAP IQ	Avg	0.50 ± 0.01	**0.51 ± 0.03**	0.52 ± 0.01	**0.55 ± 0.04**
	Linsep	0.52 ± 0.02	**0.53 ± 0.04**	0.53 ± 0.03	**0.54 ± 0.01**
	K-Means	0.50 ± 0.03	**0.51 ± 0.02**	0.52 ± 0.01	**0.55 ± 0.02**
	K-Linsep	0.52 ± 0.01	**0.53 ± 0.02**	0.53 ± 0.03	**0.54 ± 0.01**
IntGrad	Avg	0.48 ± 0.03	**0.51 ± 0.01**	0.69 ± 0.01	**0.71 ± 0.02**
	Linsep	0.49 ± 0.01	**0.52 ± 0.03**	0.67 ± 0.03	**0.71 ± 0.01**
	K-Means	**0.33 ± 0.02**	**0.33 ± 0.01**	0.34 ± 0.02	**0.35 ± 0.01**
	K-Linsep	**0.34 ± 0.01**	**0.34 ± 0.03**	**0.34 ± 0.01**	**0.34 ± 0.02**
GradCAM	Avg	0.43 ± 0.04	**0.45 ± 0.02**	**0.45 ± 0.02**	**0.45 ± 0.03**
	Linsep	0.44 ± 0.03	**0.47 ± 0.01**	0.47 ± 0.02	**0.50 ± 0.01**
	K-Means	**0.42 ± 0.03**	0.40 ± 0.02	**0.43 ± 0.02**	0.40 ± 0.01
	K-Linsep	0.43 ± 0.01	**0.45 ± 0.02**	0.44 ± 0.01	**0.47 ± 0.03**
FullGrad	Avg	0.41 ± 0.01	**0.44 ± 0.03**	0.40 ± 0.01	**0.42 ± 0.03**
	Linsep	0.44 ± 0.02	**0.45 ± 0.01**	**0.44 ± 0.02**	**0.44 ± 0.02**
	K-Means	0.37 ± 0.02	**0.42 ± 0.01**	**0.38 ± 0.03**	**0.38 ± 0.02**
	K-Linsep	**0.43 ± 0.01**	0.42 ± 0.03	0.42 ± 0.02	**0.43 ± 0.01**
CALM	Avg	**0.42 ± 0.03**	**0.42 ± 0.02**	0.44 ± 0.03	**0.45 ± 0.01**
	Linsep	0.46 ± 0.01	**0.48 ± 0.01**	0.48 ± 0.02	**0.52 ± 0.01**
	K-Means	**0.37 ± 0.03**	**0.37 ± 0.02**	0.41 ± 0.01	**0.43 ± 0.02**
	K-Linsep	0.43 ± 0.01	**0.46 ± 0.02**	0.45 ± 0.02	**0.49 ± 0.01**
MFABA	Avg	0.50 ± 0.02	**0.52 ± 0.02**	0.50 ± 0.03	**0.51 ± 0.01**
	Linsep	0.53 ± 0.02	**0.55 ± 0.03**	0.51 ± 0.01	**0.52 ± 0.02**
	K-Means	0.46 ± 0.02	**0.50 ± 0.01**	**0.48 ± 0.03**	0.47 ± 0.02
	K-Linsep	**0.51 ± 0.01**	0.49 ± 0.03	**0.49 ± 0.01**	0.47 ± 0.02

Table 7: Average F1 for the Sarcasm Dataset.

Attribution Method	Concept Type	Llama		Qwen		Gemma	
		Concept	Super Activators	Concept	Super Activators	Concept	Super Activators
CosSim	Avg	**0.39 ± 0.01**	0.25 ± 0.03	**0.38 ± 0.02**	0.26 ± 0.03	**0.42 ± 0.03**	0.25 ± 0.02
	LinSep	**0.63 ± 0.02**	0.37 ± 0.01	**0.58 ± 0.01**	0.37 ± 0.02	**0.57 ± 0.01**	0.40 ± 0.03
	K-Means	**0.28 ± 0.01**	**0.28 ± 0.03**	**0.26 ± 0.02**	0.25 ± 0.01	**0.24 ± 0.03**	0.23 ± 0.02
	K-LinSep	**0.28 ± 0.02**	**0.28 ± 0.01**	**0.24 ± 0.01**	**0.24 ± 0.03**	**0.24 ± 0.02**	0.23 ± 0.01
LIME	Avg	0.34 ± 0.01	**0.46 ± 0.03**	0.33 ± 0.03	**0.45 ± 0.01**	0.36 ± 0.02	**0.50 ± 0.01**
	LinSep	0.52 ± 0.02	**0.70 ± 0.02**	0.51 ± 0.02	**0.65 ± 0.03**	0.54 ± 0.01	**0.63 ± 0.03**
	K-Means	0.29 ± 0.01	**0.50 ± 0.02**	0.31 ± 0.02	**0.45 ± 0.01**	0.33 ± 0.01	**0.51 ± 0.02**
	K-LinSep	0.50 ± 0.03	**0.74 ± 0.01**	0.53 ± 0.01	**0.60 ± 0.03**	0.55 ± 0.03	**0.66 ± 0.01**
SHAP	Avg	0.35 ± 0.03	**0.47 ± 0.01**	0.34 ± 0.01	**0.46 ± 0.02**	0.37 ± 0.03	**0.51 ± 0.02**
	LinSep	0.53 ± 0.01	**0.71 ± 0.03**	0.52 ± 0.03	**0.66 ± 0.01**	0.55 ± 0.02	**0.64 ± 0.01**
	K-Means	0.30 ± 0.02	**0.46 ± 0.01**	0.30 ± 0.03	**0.45 ± 0.02**	0.35 ± 0.02	**0.46 ± 0.01**
	K-LinSep	0.54 ± 0.01	**0.74 ± 0.03**	0.54 ± 0.01	**0.68 ± 0.02**	0.51 ± 0.01	**0.67 ± 0.03**
RISE	Avg	0.39 ± 0.02	**0.52 ± 0.01**	0.38 ± 0.02	**0.50 ± 0.03**	0.42 ± 0.01	**0.55 ± 0.03**
	LinSep	0.57 ± 0.01	**0.76 ± 0.02**	0.56 ± 0.01	**0.71 ± 0.02**	0.59 ± 0.03	**0.69 ± 0.02**
	K-Means	0.40 ± 0.03	**0.49 ± 0.02**	0.39 ± 0.02	**0.52 ± 0.01**	0.46 ± 0.03	**0.55 ± 0.02**
	K-LinSep	0.59 ± 0.01	**0.72 ± 0.02**	0.53 ± 0.01	**0.74 ± 0.03**	0.60 ± 0.01	**0.70 ± 0.02**
SHAP IQ	Avg	0.36 ± 0.03	**0.49 ± 0.01**	0.36 ± 0.03	**0.48 ± 0.01**	0.39 ± 0.02	**0.53 ± 0.01**
	LinSep	0.55 ± 0.01	**0.73 ± 0.03**	0.54 ± 0.02	**0.68 ± 0.03**	0.57 ± 0.01	**0.66 ± 0.03**
	K-Means	0.38 ± 0.02	**0.46 ± 0.01**	0.37 ± 0.03	**0.45 ± 0.02**	0.40 ± 0.02	**0.51 ± 0.01**
	K-LinSep	0.52 ± 0.01	**0.74 ± 0.03**	0.52 ± 0.01	**0.70 ± 0.02**	0.59 ± 0.01	**0.66 ± 0.03**
IntGrad	Avg	0.27 ± 0.02	**0.40 ± 0.01**	0.27 ± 0.01	**0.39 ± 0.02**	0.29 ± 0.02	**0.43 ± 0.01**
	LinSep	0.39 ± 0.01	**0.64 ± 0.02**	0.38 ± 0.03	**0.59 ± 0.01**	0.41 ± 0.01	**0.58 ± 0.02**
	K-Means	**0.39 ± 0.03**	0.27 ± 0.02	**0.38 ± 0.02**	0.29 ± 0.01	**0.41 ± 0.03**	0.27 ± 0.02
	K-LinSep	0.38 ± 0.01	**0.67 ± 0.02**	0.41 ± 0.01	**0.58 ± 0.03**	0.39 ± 0.01	**0.58 ± 0.02**
GradCAM	Avg	0.31 ± 0.01	**0.44 ± 0.03**	0.30 ± 0.02	**0.43 ± 0.03**	0.33 ± 0.03	**0.47 ± 0.01**
	LinSep	0.43 ± 0.02	**0.68 ± 0.01**	0.42 ± 0.01	**0.63 ± 0.02**	0.45 ± 0.02	**0.62 ± 0.03**
	K-Means	0.31 ± 0.02	**0.45 ± 0.01**	0.33 ± 0.03	**0.44 ± 0.02**	0.34 ± 0.02	**0.48 ± 0.01**
	K-LinSep	0.44 ± 0.01	**0.70 ± 0.03**	0.42 ± 0.01	**0.65 ± 0.02**	0.46 ± 0.01	**0.62 ± 0.03**
FullGrad	Avg	0.28 ± 0.03	**0.41 ± 0.02**	0.28 ± 0.03	**0.40 ± 0.01**	0.30 ± 0.01	**0.44 ± 0.02**
	LinSep	0.40 ± 0.01	**0.65 ± 0.03**	0.39 ± 0.02	**0.60 ± 0.03**	0.42 ± 0.02	**0.59 ± 0.01**
	K-Means	0.28 ± 0.03	**0.39 ± 0.02**	0.26 ± 0.02	**0.43 ± 0.01**	0.29 ± 0.03	**0.41 ± 0.02**
	K-LinSep	0.38 ± 0.01	**0.65 ± 0.02**	0.41 ± 0.01	**0.58 ± 0.03**	0.42 ± 0.01	**0.60 ± 0.02**
CALM	Avg	0.34 ± 0.02	**0.47 ± 0.01**	0.33 ± 0.01	**0.46 ± 0.02**	0.36 ± 0.02	**0.50 ± 0.03**
	LinSep	0.52 ± 0.01	**0.71 ± 0.02**	0.51 ± 0.03	**0.66 ± 0.01**	0.54 ± 0.01	**0.65 ± 0.02**
	K-Means	**0.34 ± 0.02**	0.49 ± 0.01	**0.34 ± 0.03**	0.46 ± 0.02	0.36 ± 0.02	**0.49 ± 0.01**
	K-LinSep	0.51 ± 0.01	**0.72 ± 0.03**	0.50 ± 0.01	**0.67 ± 0.02**	0.56 ± 0.01	**0.66 ± 0.03**
MFABA	Avg	0.33 ± 0.03	**0.46 ± 0.01**	0.32 ± 0.02	**0.45 ± 0.03**	0.35 ± 0.03	**0.49 ± 0.01**
	LinSep	0.51 ± 0.01	**0.70 ± 0.03**	0.50 ± 0.01	**0.65 ± 0.02**	0.53 ± 0.02	**0.64 ± 0.03**
	K-Means	0.34 ± 0.03	**0.48 ± 0.02**	**0.35 ± 0.02**	0.43 ± 0.01	0.32 ± 0.03	**0.50 ± 0.02**
	K-LinSep	0.54 ± 0.01	**0.71 ± 0.02**	0.52 ± 0.01	**0.66 ± 0.03**	0.51 ± 0.01	**0.65 ± 0.02**

Table 8: Average F1 for the iSarcasm Dataset.

Attribution Method	Concept Type	Llama		Qwen		Gemma	
		Concept	Super Activators	Concept	Super Activators	Concept	Super Activators
CosSim	Avg	**0.70 ± 0.02**	0.65 ± 0.01	**0.57 ± 0.01**	0.55 ± 0.02	**0.65 ± 0.01**	0.60 ± 0.03
	LinSep	**0.81 ± 0.03**	0.74 ± 0.02	**0.74 ± 0.03**	0.65 ± 0.01	**0.83 ± 0.02**	0.71 ± 0.01
	K-Means	0.56 ± 0.02	**0.57 ± 0.01**	**0.59 ± 0.03**	**0.59 ± 0.02**	**0.60 ± 0.01**	**0.60 ± 0.03**
	K-LinSep	**0.60 ± 0.03**	**0.60 ± 0.02**	0.57 ± 0.02	**0.58 ± 0.01**	**0.60 ± 0.03**	**0.60 ± 0.02**
LIME	Avg	0.71 ± 0.02	**0.78 ± 0.01**	0.63 ± 0.02	**0.67 ± 0.03**	0.67 ± 0.03	**0.73 ± 0.02**
	LinSep	0.79 ± 0.01	**0.87 ± 0.02**	0.71 ± 0.01	**0.80 ± 0.02**	0.76 ± 0.02	**0.89 ± 0.01**
	K-Means	0.68 ± 0.03	**0.75 ± 0.01**	0.61 ± 0.02	**0.62 ± 0.03**	0.72 ± 0.01	**0.69 ± 0.02**
	K-LinSep	0.76 ± 0.02	**0.80 ± 0.03**	0.76 ± 0.01	**0.83 ± 0.02**	0.76 ± 0.02	**0.94 ± 0.01**
SHAP	Avg	0.72 ± 0.03	**0.79 ± 0.01**	0.64 ± 0.03	**0.68 ± 0.01**	0.68 ± 0.01	**0.74 ± 0.03**
	LinSep	0.80 ± 0.02	**0.88 ± 0.01**	0.72 ± 0.02	**0.81 ± 0.03**	0.77 ± 0.03	**0.90 ± 0.02**
	K-Means	0.69 ± 0.03	**0.83 ± 0.02**	0.65 ± 0.01	**0.71 ± 0.03**	0.65 ± 0.03	**0.78 ± 0.02**
	K-LinSep	0.81 ± 0.02	**0.88 ± 0.01**	0.69 ± 0.02	**0.79 ± 0.01**	0.74 ± 0.01	**0.92 ± 0.03**
RISE	Avg	0.76 ± 0.01	**0.83 ± 0.03**	0.67 ± 0.01	**0.73 ± 0.02**	0.72 ± 0.02	**0.79 ± 0.01**
	LinSep	0.84 ± 0.02	**0.92 ± 0.01**	0.76 ± 0.03	**0.85 ± 0.01**	0.81 ± 0.01	**0.94 ± 0.03**
	K-Means	**0.80 ± 0.01**	**0.80 ± 0.03**	0.64 ± 0.01	**0.75 ± 0.02**	0.74 ± 0.02	**0.81 ± 0.01**
	K-LinSep	**0.84 ± 0.03**	**0.84 ± 0.01**	0.75 ± 0.03	**0.89 ± 0.01**	0.84 ± 0.01	**0.85 ± 0.02**
SHAP IQ	Avg	0.74 ± 0.02	**0.81 ± 0.02**	0.65 ± 0.02	**0.70 ± 0.03**	0.70 ± 0.03	**0.76 ± 0.02**
	LinSep	0.82 ± 0.01	**0.90 ± 0.02**	0.74 ± 0.01	**0.83 ± 0.03**	0.79 ± 0.02	**0.92 ± 0.01**
	K-Means	0.74 ± 0.02	**0.85 ± 0.01**	0.61 ± 0.02	**0.71 ± 0.03**	0.67 ± 0.02	**0.80 ± 0.01**
	K-LinSep	0.85 ± 0.01	**0.83 ± 0.02**	0.74 ± 0.01	**0.82 ± 0.02**	0.80 ± 0.01	**0.82 ± 0.03**
IntGrad	Avg	0.66 ± 0.03	**0.71 ± 0.01**	0.56 ± 0.03	**0.58 ± 0.01**	0.61 ± 0.01	**0.66 ± 0.03**
	LinSep	0.75 ± 0.02	**0.82 ± 0.03**	0.66 ± 0.02	**0.75 ± 0.03**	0.72 ± 0.02	**0.84 ± 0.01**
	K-Means	**0.74 ± 0.01**	0.68 ± 0.03	0.56 ± 0.03	**0.53 ± 0.02**	0.65 ± 0.01	**0.63 ± 0.02**
	K-LinSep	0.75 ± 0.02	**0.74 ± 0.01**	0.66 ± 0.02	**0.77 ± 0.01**	0.74 ± 0.02	**0.88 ± 0.01**
GradCAM	Avg	0.69 ± 0.01	**0.75 ± 0.02**	0.59 ± 0.01	**0.62 ± 0.02**	0.64 ± 0.03	**0.70 ± 0.01**
	LinSep	0.78 ± 0.03	**0.86 ± 0.01**	0.69 ± 0.03	**0.78 ± 0.01**	0.74 ± 0.02	**0.87 ± 0.03**
	K-Means	0.67 ± 0.03	**0.72 ± 0.02**	0.56 ± 0.01	**0.61 ± 0.03**	0.63 ± 0.01	**0.68 ± 0.02**
	K-LinSep	0.70 ± 0.02	**0.74 ± 0.01**	0.70 ± 0.01	**0.71 ± 0.02**	0.76 ± 0.02	**0.78 ± 0.01**
FullGrad	Avg	0.67 ± 0.02	**0.72 ± 0.01**	0.57 ± 0.02	**0.60 ± 0.01**	0.62 ± 0.01	**0.67 ± 0.02**
	LinSep	0.76 ± 0.01	**0.83 ± 0.02**	0.67 ± 0.01	**0.76 ± 0.03**	0.73 ± 0.03	**0.85 ± 0.01**
	K-Means	0.66 ± 0.01	**0.73 ± 0.02**	0.56 ± 0.02	**0.63 ± 0.01**	0.61 ± 0.03	**0.65 ± 0.02**
	K-LinSep	0.73 ± 0.02	**0.82 ± 0.01**	0.64 ± 0.01	**0.75 ± 0.03**	0.70 ± 0.02	**0.87 ± 0.01**
CALM	Avg	0.71 ± 0.03	**0.78 ± 0.01**	0.61 ± 0.03	**0.66 ± 0.01**	0.66 ± 0.02	**0.73 ± 0.01**
	LinSep	0.81 ± 0.01	**0.89 ± 0.03**	0.73 ± 0.02	**0.81 ± 0.03**	0.78 ± 0.01	**0.91 ± 0.02**
	K-Means	**0.74 ± 0.03**	0.72 ± 0.02	0.61 ± 0.01	**0.64 ± 0.03**	0.66 ± 0.02	**0.65 ± 0.01**
	K-LinSep	0.80 ± 0.02	**0.82 ± 0.01**	0.72 ± 0.02	**0.73 ± 0.01**	0.75 ± 0.01	**0.79 ± 0.03**
MFABA	Avg	0.70 ± 0.02	**0.77 ± 0.01**	0.60 ± 0.02	**0.65 ± 0.01**	0.65 ± 0.03	**0.72 ± 0.01**
	LinSep	0.80 ± 0.01	**0.88 ± 0.02**	0.72 ± 0.01	**0.80 ± 0.02**	0.77 ± 0.02	**0.90 ± 0.03**
	K-Means	0.73 ± 0.01	**0.75 ± 0.02**	0.62 ± 0.01	**0.66 ± 0.03**	0.66 ± 0.03	**0.71 ± 0.02**
	K-LinSep	0.81 ± 0.02	**0.85 ± 0.01**	0.74 ± 0.02	**0.79 ± 0.01**	0.80 ± 0.01	**0.88 ± 0.03**

Table 9: Average F1 for the GoEmotions Dataset.

Attribution Method	Concept Type	Llama		Qwen		Gemma	
		Concept	Super Activators	Concept	Super Activators	Concept	Super Activators
CosSim	Avg	**0.18 ± 0.03**	0.16 ± 0.02	**0.25 ± 0.03**	0.23 ± 0.01	**0.19 ± 0.02**	0.16 ± 0.01
	LinSep	**0.29 ± 0.01**	0.25 ± 0.03	**0.31 ± 0.02**	0.28 ± 0.03	**0.25 ± 0.03**	0.23 ± 0.02
	K-Means	**0.18 ± 0.03**	**0.18 ± 0.02**	0.23 ± 0.01	**0.26 ± 0.03**	**0.15 ± 0.02**	**0.15 ± 0.01**
	K-LinSep	0.18 ± 0.01	**0.19 ± 0.03**	0.23 ± 0.03	**0.25 ± 0.02**	0.14 ± 0.01	**0.16 ± 0.03**
LIME	Avg	0.20 ± 0.03	**0.25 ± 0.01**	0.27 ± 0.01	**0.31 ± 0.02**	0.21 ± 0.01	**0.24 ± 0.03**
	LinSep	0.29 ± 0.02	**0.34 ± 0.03**	0.33 ± 0.03	**0.37 ± 0.01**	0.28 ± 0.03	**0.30 ± 0.02**
	K-Means	0.18 ± 0.02	**0.26 ± 0.01**	0.28 ± 0.02	**0.26 ± 0.03**	0.23 ± 0.02	**0.25 ± 0.01**
	K-LinSep	0.25 ± 0.01	**0.35 ± 0.02**	0.34 ± 0.01	**0.38 ± 0.02**	0.24 ± 0.01	**0.31 ± 0.03**
SHAP	Avg	0.21 ± 0.02	**0.26 ± 0.02**	0.28 ± 0.02	**0.32 ± 0.03**	0.22 ± 0.02	**0.25 ± 0.01**
	LinSep	0.30 ± 0.01	**0.35 ± 0.04**	0.34 ± 0.01	**0.38 ± 0.02**	0.29 ± 0.01	**0.31 ± 0.03**
	K-Means	0.22 ± 0.01	**0.27 ± 0.03**	0.32 ± 0.02	**0.37 ± 0.01**	0.19 ± 0.03	**0.27 ± 0.02**
	K-LinSep	0.27 ± 0.02	**0.31 ± 0.01**	0.33 ± 0.01	**0.40 ± 0.02**	0.29 ± 0.01	**0.28 ± 0.03**
RISE	Avg	0.24 ± 0.03	**0.30 ± 0.01**	0.30 ± 0.03	**0.35 ± 0.01**	0.25 ± 0.03	**0.28 ± 0.02**
	LinSep	0.33 ± 0.01	**0.39 ± 0.02**	0.37 ± 0.02	**0.42 ± 0.03**	0.32 ± 0.02	**0.35 ± 0.01**
	K-Means	0.21 ± 0.03	**0.27 ± 0.02**	0.32 ± 0.02	**0.38 ± 0.01**	0.24 ± 0.02	**0.27 ± 0.01**
	K-LinSep	**0.36 ± 0.01**	**0.36 ± 0.02**	0.37 ± 0.01	**0.42 ± 0.03**	0.32 ± 0.01	**0.34 ± 0.02**
SHAP IQ	Avg	0.22 ± 0.02	**0.28 ± 0.03**	0.29 ± 0.01	**0.33 ± 0.02**	0.23 ± 0.01	**0.26 ± 0.03**
	LinSep	0.31 ± 0.03	**0.37 ± 0.01**	0.35 ± 0.03	**0.40 ± 0.01**	0.30 ± 0.03	**0.33 ± 0.02**
	K-Means	0.20 ± 0.02	**0.27 ± 0.01**	0.28 ± 0.01	**0.31 ± 0.02**	0.24 ± 0.03	**0.22 ± 0.01**
	K-LinSep	0.34 ± 0.01	**0.35 ± 0.03**	0.35 ± 0.02	**0.38 ± 0.01**	0.29 ± 0.02	**0.35 ± 0.03**
IntGrad	Avg	0.17 ± 0.01	**0.19 ± 0.02**	0.24 ± 0.02	**0.26 ± 0.03**	0.17 ± 0.01	**0.20 ± 0.01**
	LinSep	0.26 ± 0.02	**0.30 ± 0.01**	0.29 ± 0.01	**0.32 ± 0.02**	0.24 ± 0.02	**0.26 ± 0.03**
	K-Means	0.23 ± 0.01	**0.19 ± 0.02**	0.27 ± 0.03	**0.25 ± 0.01**	0.18 ± 0.01	**0.19 ± 0.02**
	K-LinSep	0.28 ± 0.02	**0.29 ± 0.01**	0.27 ± 0.02	**0.32 ± 0.03**	0.24 ± 0.02	**0.23 ± 0.01**
GradCAM	Avg	0.19 ± 0.03	**0.23 ± 0.01**	0.26 ± 0.03	**0.29 ± 0.01**	0.19 ± 0.03	**0.22 ± 0.02**
	LinSep	0.28 ± 0.02	**0.34 ± 0.02**	0.31 ± 0.02	**0.36 ± 0.03**	0.27 ± 0.02	**0.29 ± 0.01**
	K-Means	0.20 ± 0.01	**0.21 ± 0.03**	0.25 ± 0.02	**0.31 ± 0.01**	0.20 ± 0.03	**0.21 ± 0.02**
	K-LinSep	0.27 ± 0.02	**0.34 ± 0.01**	0.33 ± 0.01	**0.35 ± 0.02**	0.25 ± 0.01	**0.26 ± 0.03**
FullGrad	Avg	0.18 ± 0.01	**0.21 ± 0.03**	0.25 ± 0.01	**0.27 ± 0.02**	0.18 ± 0.01	**0.21 ± 0.02**
	LinSep	0.27 ± 0.03	**0.31 ± 0.02**	0.30 ± 0.03	**0.33 ± 0.01**	0.25 ± 0.03	**0.27 ± 0.02**
	K-Means	0.18 ± 0.03	**0.19 ± 0.02**	0.23 ± 0.01	**0.26 ± 0.03**	0.16 ± 0.02	**0.22 ± 0.01**
	K-LinSep	0.26 ± 0.02	**0.30 ± 0.01**	0.29 ± 0.02	**0.32 ± 0.01**	0.27 ± 0.01	**0.25 ± 0.03**
CALM	Avg	0.21 ± 0.02	**0.26 ± 0.01**	0.27 ± 0.02	**0.32 ± 0.03**	0.22 ± 0.02	**0.25 ± 0.01**
	LinSep	0.30 ± 0.02	**0.36 ± 0.03**	0.34 ± 0.01	**0.39 ± 0.02**	0.29 ± 0.01	**0.32 ± 0.03**
	K-Means	0.23 ± 0.01	**0.24 ± 0.02**	0.28 ± 0.01	**0.30 ± 0.02**	0.22 ± 0.02	**0.25 ± 0.01**
	K-LinSep	0.29 ± 0.02	**0.35 ± 0.01**	0.33 ± 0.02	**0.37 ± 0.01**	0.27 ± 0.01	**0.30 ± 0.03**
MFABA	Avg	0.20 ± 0.01	**0.25 ± 0.03**	0.26 ± 0.03	**0.31 ± 0.01**	0.21 ± 0.03	**0.24 ± 0.01**
	LinSep	0.29 ± 0.02	**0.35 ± 0.01**	0.33 ± 0.02	**0.38 ± 0.03**	0.28 ± 0.02	**0.31 ± 0.03**
	K-Means	0.19 ± 0.01	**0.26 ± 0.03**	0.27 ± 0.02	**0.34 ± 0.01**	0.23 ± 0.02	**0.26 ± 0.01**
	K-LinSep	0.28 ± 0.02	**0.36 ± 0.01**	0.32 ± 0.01	**0.36 ± 0.03**	0.29 ± 0.03	**0.34 ± 0.02**

L Sparse Autoencoders

L.1 SAEs for Concept Detection

Sparse autoencoders [20] (SAEs) are mechanism for uncovering latent concepts in large models. By training an encoder–decoder architecture with sparsity constraints, SAEs aim to discover a set of basis features that are both interpretable and disentangled. This approach is attractive for concept analysis because sparsity encourages individual hidden units to capture relatively specific and semantically meaningful directions in representation space. In principle, such units could act as natural "concept detectors" without additional supervision.

Despite these benefits, SAEs come with notable limitations. Training them at scale is extremely resource-intensive, and thus only a small number of pretrained SAEs have been made publicly available. These models are typically trained on very specific layers of particular architectures and cannot be easily transferred to other checkpoints or layers. For this reason, we restrict our comparisons to what is currently feasible: an SAE trained on the penultimate residual stream of CLIP [43, 87, 88] (covering 92% of the model depth for images) and SAEs trained on intermediate layers of Gemma [45, 89] (covering 81% of the depth for text). A second practical issue is that SAEs output thousands of candidate units, which makes automatic labeling more difficult. To address this, we filtered out units that activated on nearly all samples or no samples [90], or with insufficient activation strength [91].

After filtering, we evaluated the retained SAE units as potential unsupervised concept detectors. We apply the same SuperActivator paradigm for detection, treating [CLS] and token-alignment with the retained SAE units as concept activation scores.

Table 10 shows the F_1 concept detection performance for the best-perfoming SAE units for each ground truth concept. Our SuperActivators method performs quite well across all datasets. However, we note in Figure 36 that our method achieved peak performance by just using a much larger subset of the most activated tokens (larger δ). We suspect this is due to the sparsity constraint in SAE training objectives. By penalizing high activations, SAEs eliminate weak and noisy responses and shrink the scale of the surviving ones. With less contrast between the strongest and moderate responses, concept evidence becomes spread across more activated tokens and less concentrated in the tail.

Table 10: Detection F1 (avg. across concepts) from SAE concepts: 92% through *CLIP* for image datasets and 81% through *Gemma* for text datasets.

	Concept Detection Methods				
	CLS	RandTok	LastTok	MeanTok	SuperTok (Ours)
CLEVR	0.898 ± 0.135	0.504 ± 0.077	0.504 ± 0.077	0.609 ± 0.083	**0.992 ± 0.090**
COCO	0.462 ± 0.064	0.335 ± 0.049	0.339 ± 0.049	**0.591 ± 0.069**	0.582 ± 0.000
Surfaces	0.419 ± 0.062	0.345 ± 0.042	0.344 ± 0.042	0.479 ± 0.074	**0.501 ± 0.085**
Pascal	0.570 ± 0.063	0.398 ± 0.049	0.404 ± 0.053	0.601 ± 0.060	**0.662 ± 0.000**
Sarcasm	**0.662 ± 0.075**	0.659 ± 0.052	0.659 ± 0.052	0.659 ± 0.052	0.659 ± 0.052
iSarcasm	0.706 ± 0.069	0.676 ± 0.044	0.676 ± 0.044	0.703 ± 0.051	**0.777 ± 0.054**
GoEmotions	0.159 ± 0.067	0.124 ± 0.062	0.124 ± 0.062	0.350 ± 0.106	**0.395 ± 0.093**

L.2 SAEs for Concept Attribution

Having established that SAEs can act as competitive unsupervised detectors, we next evaluate whether they can also support concept attribution. Tables 11 and 12 report average attribution F_1 across both image and text datasets.

Across all methods, we observe a consistent pattern: using the average of local SuperActivators derived from SAE concepts produces attribution maps that align better with ground truth labels and score higher on faithfulness. On image datasets, SuperActivators improves scores in nearly every setting, often by non-trivial margins. Similar trends appear in text, where SuperActivators again provides the strongest performance in most cases.

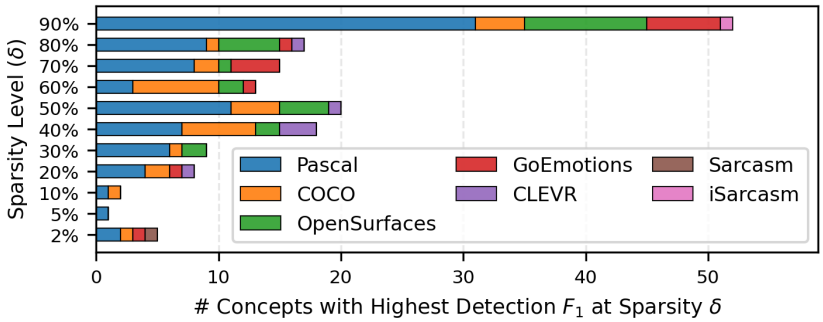


Figure 36: For SAEs The strongest globally applicable concept signals are not concentrated in a very sparse set of signals.

While the average F_1 across all concepts remains modest relative to supervised baselines, the results highlight a consistent trend: even for SAEs, SuperActivators consistently provides a more accurate signal for both concept detection and attribution than global CLS-based pooling. This suggests that fine-grained, token-level alignment is crucial for extracting interpretable signals from unsupervised representations.

Table 11: Average Attribution F1 for SAEs on Image Datasets with CLIP model.

(a) CLEVR and COCO Dataset

Attribution Method	CLEVR		COCO	
	CLS	SuperActivators	CLS	SuperActivators
LIME	0.45 ± 0.04	$\mathbf{0.49 \pm 0.01}$	0.32 ± 0.03	$\mathbf{0.33 \pm 0.04}$
SHAP	0.47 ± 0.05	$\mathbf{0.51 \pm 0.03}$	0.31 ± 0.03	$\mathbf{0.34 \pm 0.02}$
RISE	0.44 ± 0.03	$\mathbf{0.48 \pm 0.03}$	0.30 ± 0.02	$\mathbf{0.33 \pm 0.01}$
SHAP IQ	$\mathbf{0.46 \pm 0.04}$	$\mathbf{0.46 \pm 0.02}$	0.28 ± 0.05	$\mathbf{0.33 \pm 0.04}$
IntGrad	0.40 ± 0.05	$\mathbf{0.44 \pm 0.04}$	0.27 ± 0.04	$\mathbf{0.31 \pm 0.03}$
GradCAM	0.36 ± 0.05	$\mathbf{0.40 \pm 0.05}$	0.26 ± 0.05	$\mathbf{0.30 \pm 0.04}$
FullGrad	0.37 ± 0.04	$\mathbf{0.41 \pm 0.02}$	$\mathbf{0.32 \pm 0.03}$	0.31 ± 0.04
CALM	0.44 ± 0.02	$\mathbf{0.49 \pm 0.04}$	0.27 ± 0.05	$\mathbf{0.32 \pm 0.03}$
MFABA	0.44 ± 0.03	$\mathbf{0.49 \pm 0.02}$	0.28 ± 0.04	$\mathbf{0.30 \pm 0.03}$

(b) OpenSurfaces and Pascal Dataset

Attribution Method	OpenSurfaces		Pascal	
	CLS	SuperActivators	CLS	SuperActivators
LIME	0.41 ± 0.04	$\mathbf{0.43 \pm 0.04}$	0.40 ± 0.05	$\mathbf{0.44 \pm 0.04}$
SHAP	0.31 ± 0.03	$\mathbf{0.35 \pm 0.02}$	0.41 ± 0.04	$\mathbf{0.45 \pm 0.03}$
RISE	0.36 ± 0.05	$\mathbf{0.40 \pm 0.02}$	0.40 ± 0.05	$\mathbf{0.44 \pm 0.05}$
SHAP IQ	0.37 ± 0.04	$\mathbf{0.41 \pm 0.05}$	0.41 ± 0.05	$\mathbf{0.45 \pm 0.01}$
IntGrad	0.39 ± 0.02	$\mathbf{0.43 \pm 0.02}$	0.46 ± 0.05	$\mathbf{0.50 \pm 0.02}$
GradCAM	0.32 ± 0.05	$\mathbf{0.36 \pm 0.02}$	0.34 ± 0.03	$\mathbf{0.38 \pm 0.04}$
FullGrad	0.34 ± 0.03	$\mathbf{0.38 \pm 0.03}$	0.36 ± 0.05	$\mathbf{0.40 \pm 0.02}$
CALM	0.26 ± 0.05	$\mathbf{0.30 \pm 0.02}$	0.35 ± 0.04	$\mathbf{0.39 \pm 0.03}$
MFABA	$\mathbf{0.39 \pm 0.04}$	$\mathbf{0.39 \pm 0.02}$	0.41 ± 0.03	$\mathbf{0.46 \pm 0.02}$

Table 12: Average Attribution F1 for SAEs on Text Datasets with Gemma Model.

Attribution Method	Sarcasm		iSarcasm		GoEmotions	
	CLS	Super Activators	CLS	Super Activators	CLS	Super Activators
LIME	**0.37 ± 0.05**	0.36 ± 0.02	0.62 ± 0.03	**0.65 ± 0.04**	0.16 ± 0.04	**0.20 ± 0.04**
SHAP	0.33 ± 0.04	**0.37 ± 0.04**	0.59 ± 0.05	**0.64 ± 0.01**	0.18 ± 0.03	**0.23 ± 0.02**
RISE	0.37 ± 0.05	**0.42 ± 0.03**	0.68 ± 0.04	**0.72 ± 0.04**	0.20 ± 0.05	**0.22 ± 0.02**
SHAP IQ	**0.40 ± 0.05**	**0.40 ± 0.02**	0.68 ± 0.05	**0.69 ± 0.02**	0.18 ± 0.04	**0.23 ± 0.02**
IntGrad	0.31 ± 0.05	**0.35 ± 0.04**	0.52 ± 0.05	**0.57 ± 0.04**	0.10 ± 0.04	**0.15 ± 0.05**
GradCAM	0.34 ± 0.04	**0.39 ± 0.03**	0.53 ± 0.03	**0.58 ± 0.01**	0.16 ± 0.05	**0.20 ± 0.02**
FullGrad	0.28 ± 0.05	**0.33 ± 0.03**	**0.59 ± 0.04**	**0.59 ± 0.03**	0.14 ± 0.03	**0.18 ± 0.04**
CALM	0.37 ± 0.04	**0.39 ± 0.04**	0.56 ± 0.05	**0.60 ± 0.04**	0.16 ± 0.03	**0.21 ± 0.02**
MFABA	0.33 ± 0.03	**0.38 ± 0.03**	0.55 ± 0.04	**0.60 ± 0.02**	0.18 ± 0.03	**0.23 ± 0.02**