Constructing Interpretable Features from Compositional Neuron Groups

Or Shafran[1] **Atticus Geiger**[2] **Mor Geva**[1]

[1]Blavatnik School of Computer Science and AI, Tel Aviv University [2]Pr(Ai)[2]R Group

{ordavids1@mail, morgeva@tauex}.tau.ac.il, atticusg@gmail.com

Abstract

A central goal for mechanistic interpretability has been to identify the right units of analysis in large language models (LLMs) that causally explain their outputs. While early work focused on individual neurons, evidence that neurons often encode multiple concepts has motivated a shift toward analyzing directions in activation space. A key question is how to find directions that capture interpretable features in an unsupervised manner. Current methods rely on dictionary learning with sparse autoencoders (SAEs), commonly trained over residual stream activations to learn directions from scratch. However, SAEs often struggle in causal evaluations and lack intrinsic interpretability, as their learning is not explicitly tied to the computations of the model. Here, we tackle these limitations by directly decomposing MLP activations with semi-nonnegative matrix factorization (SNMF), such that the learned features are (a) sparse linear combinations of co-activated neurons, and (b) mapped to their activating inputs, making them directly interpretable. Experiments on Llama 3.1, Gemma 2 and GPT-2 show that SNMF derived features outperform SAEs and a strong supervised baseline (difference-in-means) on causal steering, while aligning with human-interpretable concepts. Further analysis reveals that specific neuron combinations are reused across semantically-related features, exposing a hierarchical structure in the MLP's activation space. Together, these results position SNMF as a simple and effective tool for identifying interpretable features and dissecting concept representations in LLMs.

1 Introduction

A central question for mechanistic interpretability is how to decompose hidden representations of large language models (LLMs) into constituent parts (Mueller et al., 2024; Geiger et al., 2024; Sharkey et al., 2025). A natural widely-studied unit is individual neural activations. However, while individual activations often demonstrate human-interpretable concepts (Karpathy et al., 2015; Geva et al., 2021, 2022; Choi et al., 2024), it has long been established that they participate in the representation of multiple intelligible concepts (Smolensky, 1986; Rumelhart et al., 1986; Olah et al., 2020; Bolukbasi et al., 2021; Gurnee et al., 2023).

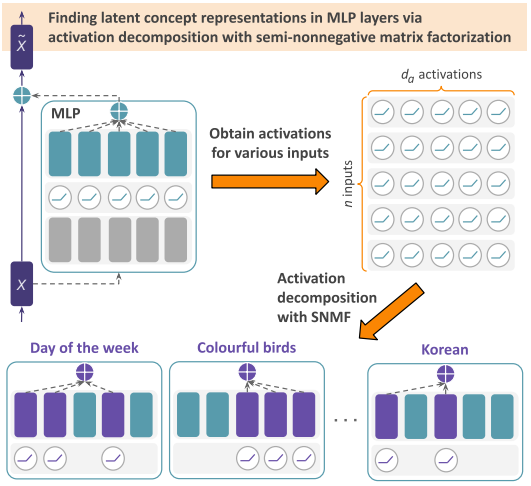

Figure 1: We find latent concept representations by decomposing MLP activations with semi-nonnegative matrix factorization (SNMF). Our method finds sparse compositional MLP features that align with interpretable concepts and outperform existing unsupervised methods and a strong supervised baseline at causal steering.

An alternative unit of analysis that has recently gained consensus is directions in activation space (Elhage et al., 2021; Gurnee et al., 2023; Nanda et al., 2023; Park et al., 2024). A key question is how to find directions encoding interpretable features in an unsupervised manner, namely, without any prior about the concepts they represent. A large body of work has explored dictionary learning methods with sparse autoencoders (SAEs; Bricken et al. 2023; He et al. 2024; Lieberum et al. 2024) to learn directions called *features*. Features are then assigned textual labels with automatic pipelines (Hernandez et al., 2021; Bills et al., 2023; Lee

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 42326–42348

et al., 2023; Gur-Arieh et al., 2025). However, while SAEs can extract interpretable features, they often fail to provide a better unit of analysis in causal evaluations (Huang et al. 2024; Chaudhary and Geiger 2024; Wu et al. 2025; Mueller et al. 2025; cf. Marks et al. 2025). Moreover, the learned features are not constrained to the model's representation space (Menon et al., 2025; Paulo and Belrose, 2025) nor grounded in specific model mechanisms.

In this work, we tackle the problem of finding interpretable features in LLMs by *decomposing MLP activations*. Instead of learning directions from scratch, as in widely-used SAEs trained on residual stream representations, our method represents features as *sparse linear combinations of existing directions in MLP weight matrices*, defined by groups of neurons (see Fig. 1). To decompose MLP activations, we propose using *semi-nonnegative matrix factorization* (SNMF; Ding et al. 2010). We collect MLP activations with d_a dimensions from a large n-token text, and decompose the matrix $A \in \mathbb{R}^{d_a \times n}$ of data into two matrices: $Z \in \mathbb{R}^{d_a \times k}$ of MLP features that define sparse neuron combinations, and a nonnegative coefficient matrix $Y \in \mathbb{R}^{k \times n}_{\geq 0}$. A fundamental advantage of this approach over SAEs is that the coefficient matrix Y encodes which tokens contributed to the creation of a feature.

We use texts that strongly activate features to assign textual *concept labels*, and evaluate features along two axes (Huang et al., 2023; Wu et al., 2025): (1) *concept detection*, measuring the ability of the feature to separate texts exemplifying the concept label and neutral texts, and (2) *concept steering*, evaluating how well feature activation steers generated texts toward the concept label while preserving fluency. We find that features learned with SNMF outperform state-of-the-art SAEs, Gemmascope and Llamascope (Lieberum et al., 2024; He et al., 2024), and supervised features computed with difference-in-means (Marks and Tegmark, 2024; Rimsky et al., 2024; Turner et al., 2024) on concept steering, while performing comparably on concept detection.

Further analysis shows that MLP features are compositional: recursively applying SNMF to the learned MLP features further composes features of fine-grained concepts (e.g., *Javascript* and *Python*) into features corresponding to more high-level concepts (e.g., *programming*). This shows the inverse of the feature splitting phenomenon observed in SAEs, where larger SAEs have clusters of features corresponding to individual features of smaller SAEs (Bricken et al., 2023; Chanin et al., 2024). Moreover, semantically-related features (e.g., the different weekdays) share a core set of neurons and vary by exclusive neurons that control the promotion and suppression of fine-grained concepts.

To conclude, we propose decomposing MLP activations with SNMF to find interpretable features in LLMs. Our experiments show the effectiveness of this approach compared to existing unsupervised methods and a strong supervised baseline, and our analysis provides insights into how neuron compositionality constructs features in MLP layers. We publicly release the code at https://github.com/ordavid-s/snmf-mlp-decomposition.

2 Related Work

Analyzing Neurons in Transformers Geva et al. (2021, 2022) view the MLP output as a collection of updates to the residual stream, where updates correspond to single neurons that often encode concepts that are human-interpretable. Alternatively, Meng et al. (2022) considers an MLP layer as an associative memory, where all neural activations form a key that induces the layer's output. Recently, Cao et al. (2025) used clustering to identify meaningful groups of neurons in CNNs. Dalvi et al. (2018); Gurnee et al. (2023); Oikarinen and Weng (2024) used supervised methods to isolate groups of neurons in transformers that are responsible for promoting specific concepts or tasks. Unlike prior work, we propose an unsupervised approach that identifies features via patterns in MLP activations. We view neuron combinations as causal mediators reused across inputs to represent similar concepts, extending prior work by showing that such reuse is a natural mechanism the model employs to represent concepts.

Concept Representations in LLMs Recent work (Marks and Tegmark, 2024; Park et al., 2024; Nanda et al., 2023) proposes the linear representation hypothesis; concept representations in language models are approximately linear, aligned with specific directions in the residual stream. However, several works challenge this view (Csordás et al., 2024; Park et al., 2025; Engels et al., 2025; Levy and Geva, 2025), proposing that some concepts have meaning when viewed in multiple dimensions. These works demonstrate the presence of structured, complex concept representations. However, they do not explain how such representations are formed or grounded in the model's under-

lying mechanisms. Our work applies SNMF to investigate how neurons form complex features, suggesting a complementary view that defines multi-dimensional concepts through their formation.

Finding Latent Concepts Supervised methods have been shown effective in finding directions in activation space that localize (Geiger et al., 2023; Wu et al., 2023) or steer (Wu et al., 2024; Marks and Tegmark, 2024) predefined concepts. However, fewer works propose unsupervised methods for finding meaningful directions. Earlier work identified salient neurons and neuron groups through probing, clustering, concept alignment, and causal analysis (Lakretz et al., 2019; Mu and Andreas, 2020; Dai et al., 2022), but these approaches generally rely on targeted analyses tied to predefined concepts or neuron-level attribution rather than recovering features directly from activation structure itself. More recently, SAEs have been used to learn directions that align with interpretable concepts (Bricken et al., 2023; Cunningham et al., 2023). Unlike SAEs, our work uses SNMF to construct features through direct matrix factorization of MLP activations, tying each feature directly to the model components responsible for forming it and representing it as a linear combination of neurons.

NMF and Deep Learning NMF has previously been used to identify interpretable concepts in deep neural networks. Prior work (Lee and Seung, 1999; Melas-Kyriazi et al., 2022; Fel et al., 2023; Jourdan et al., 2023; Oldfield et al., 2023) has demonstrated NMF isolates meaningful parts in images, e.g., eyes and noses, by factorizing image data. Yun et al. (2021) applied NMF-based dictionary learning to contextualized transformer embeddings across layers, using the learned factors mainly for linguistic analysis. In contrast, we apply SNMF to MLP activations to recover features defined by groups of co-activated neurons, directly linking each feature to the model components that realize it and evaluating these features through causal steering.

3 Decomposing MLP Activations with Semi-Nonnegative Matrix Factorization

We propose an unsupervised method for finding meaningful directions in LLMs, focusing on MLP layers. Our approach is motivated by the view that the MLP output to the residual stream can be expressed as a linear combination of underlying features, each formed by a specific set of neurons. Their activations specify how they are linearly combined to construct concepts in a given input. Thus, inputs that express similar concepts activate similar neurons. This theory is further supported by Yun et al. (2021) who provide evidence of the additive nature of concepts in the model's residual stream.

Following this view, we propose to identify groups of neurons representing interpretable features using semi-nonnegative matrix factorization (Ding et al., 2010). SNMF factorizes a data matrix into two matrices: a matrix of *features* and a nonnegative *coefficient matrix* that maps between inputs and features. Our choice of SNMF is motivated by Lee and Seung (1999) who demonstrated that the nonnegativity constraints in NMF encourage parts-based, additive representations, which tend to yield interpretable features. SNMF retains this interpretability while relaxing the constraint for nonnegativity in the features, thus being a better fit for MLP activations, which often include both positive and negative values across architectures.

Next, we explain how we apply SNMF to find interpretable features in MLP layers.

MLP Activations We assume a transformer-based language model with L layers, a hidden dimension d, and an MLP inner-dimension d_a. An MLP layer is defined with two parameter matrices $W_K, W_V^T \in \mathbb{R}^{d_a \times d}$ and an element-wise nonlinear activation function σ:[1]

$$\texttt{MLP}(\mathbf{h}) = W_V \sigma(W_K \mathbf{h}) := W_V \mathbf{a} \quad (1)$$

where $\mathbf{h} \in \mathbb{R}^d$ is an input hidden representation and $\mathbf{a} \in \mathbb{R}^{d_a}$ is a vector of activations.[2] Let $\mathbf{k}_i$ be the i-th row of W_K and $\mathbf{v}_i$ the i-th column of W_V, the output of the MLP layer can be cast as a linear combination of W_V's columns, each weighted by its corresponding activation:

$$\texttt{MLP}(\mathbf{h}) = \sum_{i=1}^{d_a} \sigma(\mathbf{k}_i \cdot \mathbf{h}) \mathbf{v}_i = \sum_{i=1}^{d_a} a_i \mathbf{v}_i \quad (2)$$

Decomposing Activations with SNMF Given neuron activations $A := [\mathbf{a}_1 \dots \mathbf{a}_n] \in \mathbb{R}^{d_a \times n}$ obtained from an MLP layer for a set of n corresponding input representations $\mathbf{h}_1, ..., \mathbf{h}_n$, we apply SNMF to decompose A into MLP features. Specifically, A is factorized into a product of two matrices $Z \in \mathbb{R}^{d_a \times k}$ and $Y \in \mathbb{R}^{k \times n}_{\geq 0}$ such that:

$$A \approx ZY, \quad (3)$$

[1] Bias terms are omitted for brevity.

[2] In modern LLMs, activations often go through additional gating before the output projection (Liu et al., 2021).

where k is a hyperparameter defining the number of features.

The columns of Z define the representations of the *MLP features*. Each MLP feature $\mathbf{z}_i \in \mathbb{R}^{d_a}$ lies in the MLP activation space and specifies a linear combination of neurons. Since the rows of A are neuron activations, each MLP feature $\mathbf{z}_i$ captures a co-activation pattern across neurons, so multiplying by W_V maps it to the residual stream defining a *residual stream feature* $\mathbf{f}_i \in \mathbb{R}^d$:

$$\mathbf{f}_i = W_V \mathbf{z}_i = \sum_{j=1}^{d_a} z_{i,j} \mathbf{v}_j \tag{4}$$

We encourage these linear combinations to be *sparse*, as typically observed in MLP layers in LLMs (Geva et al., 2021) (see details below). The *coefficient matrix* Y is restricted to nonnegative values, which specify how each input representation $\mathbf{h}_i$ is reconstructed using the MLP features. Each entry $Y_{i,j}$ indicates how strongly MLP feature i contributes to sample j. Together, Y and Z provide a parts-based decomposition, where the original activations are reconstructed as additive combinations of the MLP features.

Optimization To learn the decomposition, we initialize the matrices Z and Y such that the entries of Z are drawn from $\mathcal{U}(0,1)$ and the entries of Y are drawn from $\mathcal{N}(0,1)$. We compare additional initialization strategies in §A. Then we use the Multiplicative Updates scheme of Ding et al. (2010), which alternates between a closed-form update of Z and a multiplicative update of the nonnegative matrix Y to minimize the Frobenius norm between A and its reconstruction ZY. Lastly, we impose sparsity by applying a hard winner-take-all operator to every column of Z, keeping only the largest $p\%$ of entries (by absolute value) and zeroing the remainder as suggested in Peharz and Pernkopf (2012). We report the effect of varying p in §A.

4 Concept Detection Evaluations

We start by evaluating if the features learned by SNMF capture generalizable input-dependent patterns. We show that the MLP features consistently activate in response to concept-related inputs and remain inactive for neutral inputs, suggesting that a stable group of neurons underlies each concept. Moreover, they perform comparably to those of large-scale SAEs trained on MLP outputs and better than matched-capacity SAEs, supporting their relevance to the model's input.

4.1 Experiment

We follow previous work (Paulo et al., 2024; Gur-Arieh et al., 2025; Wu et al., 2025) and measure the ability of MLP features to discern between contexts that exemplify the concept and those that do not. Concretely, we first describe the concept captured by an MLP feature based on inputs that activate it, and then evaluate if the feature activates on inputs that were generated based on the description.

To describe the concept captured by MLP features, we leverage GPT-4o-mini (OpenAI et al., 2024) and prompt it to describe prominent patterns in the inputs that most strongly activate it (Bills et al., 2023; Bricken et al., 2023; Choi et al., 2024; Paulo et al., 2024) (see §H.2 for the prompt). Then, for every description we generate five activating sentences that express the concept and five neutral sentences that do not. Next, we run each sentence through the model and record the MLP activations for every token during the forward pass.

To evaluate how well an MLP feature aligns with its description, we compute the cosine similarity between the feature and each token activation in a sentence, and take the maximum similarity per sentence.[3] We then average these maximum scores across the activating sentences to obtain $\bar{a}_{\text{activating}}$, and across the neutral sentences to obtain $\bar{a}_{\text{neutral}}$. Lastly, we measure the log-ratio between the mean scores for the activating and neutral sentences to obtain the Concept Detection score (S_{CD}):

$$S_{CD} := \log \frac{\bar{a}_{\text{activating}}}{\bar{a}_{\text{neutral}}} \tag{5}$$

We use logarithm to normalize the scale for comparability across methods. A positive log-ratio indicates greater similarity with activating sentences, where larger values indicate better separability.

We apply this evaluation to SNMF-learned MLP features with $k = 100, 200, 300, 400$ on multiple layers of Gemma-2-2B (Team et al., 2024), Llama-3.1-8B (Grattafiori et al., 2024), and GPT-2 Small (Radford et al., 2019). Additional details on the training of SNMF are in §D. As the main baseline, we use state-of-the-art SAEs, Llamascope (He et al., 2024) and Gemmascope (Lieberum et al., 2024), trained on MLP outputs (SAE out), i.e. $W_V \mathbf{a}$ (Eq. 3). Additionally, to highlight method differences, we evaluate matched-capacity SAEs trained on MLP activations (SAE act) with the

[3] We use cosine similarity as opposed to projection to make the scales of different methods comparable (see details in §B).

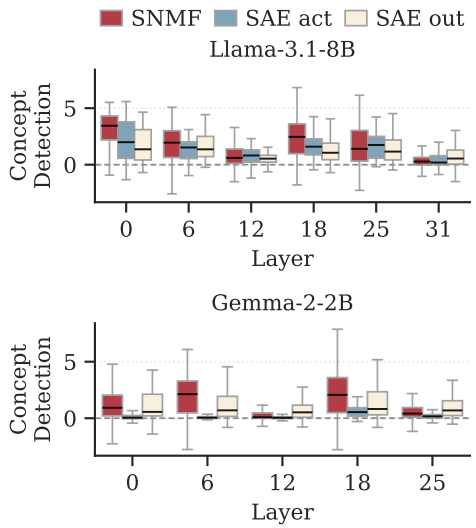

Figure 2: The distribution of concept detection scores across selected layers in Llama-3.1-8B and Gemma-2-2B for SNMF (k=100), SAE trained on MLP activations with the same dataset as SNMF (SAE act) and publicly-available SAEs trained on MLP outputs (SAE out).

same dataset and number of features as SNMF. For additional details on training the matched-capacity baseline see §C. For SNMF, we generate concept labels from the input samples that most strongly activate it, using the coefficient matrix Y. For SAE act, we take the feature descriptions from Neuronpedia, which are derived using autointerp (Paulo et al., 2024) and similarly rely on the highest-activating examples. For SAEs trained on the SNMF dataset, we generate descriptions using their top-activating contexts with the same prompts used for SNMF.

4.2 Results

Fig. 2 shows the Concept Detection score for SNMF MLP features, SAE act MLP features and SAE out features in Llama-3.1-8B and Gemma-2-2B. For SNMF we present the results for $k = 100$, which shows similar trends to other values of k. Additional results showing similar trends for GPT-2 Small and comparisons across different k values are in §D. Across different layers and models, most of the SNMF MLP features (>75%) obtain a positive concept detection score, indicating that most SNMF-discovered features activate more for concept-related inputs than for neutral ones. This suggests that MLP features are meaningfully tied to the appearance of specific concepts in the input. Interestingly, we observe comparatively high scores in the first layer of both Llama-3.1-8B and GPT-2. We hypothesize that this is because the activation representations have passed through fewer attention layers so they encode fewer entangled concepts, making them easier to decompose.

Concept	Top Activating Contexts

The word "resonate"	`... nostalgia and innovation that resonates`
	`... captivates the eye, resonating with the`
	`... campaign that would resonate more strongly`

Implementing / establishing	`the police enacted curfews and employed sound cannons`
	`... establishing a coherent code`
	`schools must implement a comprehensive framework ...`

Historical documentation	`stone carvings, striving to reveal the profound stories ...`
	`... historical narratives preserved by the ancients`
	`... texts found in the dusty archives`

Table 1: Example concepts discovered by SNMF in Llama-3.1-8B, along with their top activating contexts extracted from the coefficient matrix Y.

When comparing methods, SNMF generally achieves concept detection scores that are comparable to or exceed those of existing SAEs trained on MLP outputs and a better performance compared to capacity-matched baseline SAEs trained on the SNMF dataset. A minority of MLP features receive negative scores, indicating they respond more strongly to neutral contexts, possibly reflecting sensitivity to noise or layer-specific biases.

These results show that SNMF reliably identifies sets of neurons that correspond to human-interpretable concepts. Notably, the higher concept detection scores for SNMF may stem from its improved interpretability provided by the coefficient matrix Y, which associates between features and their activating contexts. Table 1, presents examples of identified concepts and the tokens most strongly associated with them according to Y.

5 Concept Steering Evaluations

In this section, we show that MLP features match and often exceed the steering performance of strong unsupervised and supervised baselines. Together with the interpretability results, these findings show the success of our approach in isolating meaningful neuron combinations used by the model.

5.1 Experiment

We evaluate whether MLP features discovered by SNMF causally influence the model's output in a concept-consistent manner (Paulo et al., 2024; Gur-Arieh et al., 2025; Wu et al., 2025). To this end, we

prompt the model with "<BOS> I think that", while steering with a feature during inference. To control the steering strength, we perform a grid search over seven target values of KL divergence between the output logits of a standard forward pass and of an intervened pass where the feature is amplified (Paulo et al., 2024). As some features may suppress a concept rather than promote it, we consider both the feature $\mathbf{f}$ and its negation $-\mathbf{f}$. For each KL divergence value and sign, we sample 8 generations, to obtain a total of 112 completions.

Next, we prompt GPT-4o-mini (OpenAI et al., 2024) to score each generated continuation along two axes: *concept expression* (concept score) and *fluency* (fluency score).[4] We use the evaluation prompts by Wu et al. (2025), where each axis is scored from 0-2. For each feature, we take the best result over KL/sign combinations. We report two scores: (a) the concept score, which evaluates how well the generated completions align with the target concept, and (b) the harmonic mean of the concept and fluency scores, which quantifies how effectively a feature can steer the model's generation toward the target concept while preserving output coherence. Fluency scores are less informative in isolation and are reported in §E.2.

We benchmark SNMF against Gemmascope and Llamascope (SAE out), the matched-capacity baseline (SAE act), and a strong supervised baseline, Difference-in-Means (DiffMeans; Wu et al., 2025; Rimsky et al., 2024; Marks and Tegmark, 2024; Turner et al., 2024). We do not include classical methods, such as PCA and ICA, as prior work (Wu et al., 2025; Cunningham et al., 2023) show they underperform SAEs in causal settings. Concept scores are computed with respect to a reference feature description. Gur-Arieh et al. (2025) showed that input-centric descriptions are better at describing features in early layers, while output-centric descriptions are better for describing features in upper layers. Therefore, we generate descriptions for early layer features using activating inputs (as explained in §4). For later layers, across all methods, we use an output-centric description. For an MLP feature (SNMF and SAE act), $\mathbf{z}_i$, we calculate its projection onto the residual stream via the MLP out weights,

$$\mathbf{f}_i := W_V \mathbf{z}_i \tag{6}$$

For features of SAE out and DiffMeans we directly use $\mathbf{f}_i$ as they were trained on MLP outputs. Next, we project the feature to the model's vocabulary space using the unembedding matrix, and generate the description based on the set of tokens corresponding to the top and bottom logits in the projection (Gur-Arieh et al., 2025).

We apply this evaluation to SNMF features with $k = 100$ across multiple layers in LLaMA-3.1–8B and Gemma-2–2B. For SAE act we use all 100 features. For SAE out, we randomly sample 100 features from Gemmascope and LLamascope and use the respective decoder rows. Since DiffMeans is supervised, we learn features for the concepts identified by SNMF and SAEs. For each concept, we use its description to generate 72 activating and 72 neutral sentences (as in §4) and define the feature as the difference between the average token representations for each set (Wu et al., 2025).

5.2 Results

Fig. 3 presents the results, showing varying performance across the layers of both models. Early layers tend to show lower final scores but relatively high concept scores. This discrepancy arises from the sensitivity of early-layer steering. Modifications at these layers propagate through the network, often strongly influencing model fluency and as such impacting the final score.

Notably, SNMF consistently outperforms SAEs across all layers and often matches or exceeds the performance of DiffMeans, a supervised baseline. Since DiffMeans computes feature directions by subtracting positive and negative representations, it is susceptible to noise by unrelated concepts present in the positive samples. This issue is especially prominent in earlier layers, where limited contextualization leaves token representations within the positive sample distinct, making their average less meaningful. In contrast, SNMF produces a parts-based decomposition that captures structure consistently associated with the target concept, leading to improved performance across all layers.

These results demonstrate that SNMF is able to isolate groups of neurons encoding concepts, supporting its viability as an interpretable unsupervised method for discovering meaningful directions through the MLP. Furthermore, the ability of MLP features produced by SNMF to reliably steer model outputs implies that the MLP operates using additive updates composed of interpretable neuron sets, which extends the findings of Geva et al. (2022) that single MLP vectors promote concepts in the

[4] We omit the instruct score used in Wu et al. (2025) as we evaluate base models rather than instruction tuned.

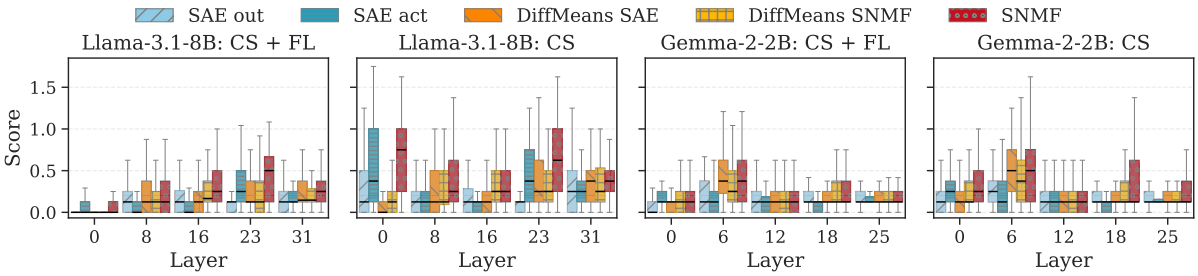

Figure 3: Causality evaluation results across layers in Llama-3.1-8B and Gemma-2-2B, using publicly-available SAEs (`SAE out`), SAE trained on MLP activations (`SAE act`), `SNMF`, and `DiffMeans` trained on concepts detected by SAE or `SNMF`. Concept Steering + Fluency (CS + FL) measure steering performance while preserving coherence, and Concept Steering (CS) quantifies how strongly the target concept is steered. Across both models `SNMF` scores often exceed both `DiffMeans` and SAEs, showing strong causal influence.

residual stream. Each MLP feature corresponds to a distinct combination of neurons whose activations affect the residual stream in ways aligned with human-interpretable concepts. Considering that neuron combinations activate to represent a concept, it follows naturally that a single neuron may participate in multiple concepts, and that a single concept may activate multiple neurons.

6 Analyzing Neuron Compositionality

Our experiments show that `SNMF` finds groups of neurons that represent interpretable features. A natural question that arises is *how are different neurons combined to form features?* Here, we investigate this question using the properties of `SNMF`.

Feature Merging via Recursive SNMF We apply `SNMF` recursively with progressively smaller values of k, which encourage more generic patterns (Song and Lee, 2013). Namely, we start by decomposing the activations and then recursively decompose the generated MLP features. This process reveals subsets of neurons that are shared across features, forming a hierarchy that shows how the model additively combines groups of neurons to represent concepts. We run this experiment on a dataset of sentences related to the concept of time units (e.g., months, weekdays, minutes) and use the coefficient matrices Y from each step to identify relationships between MLP features. More details on the recursive application of `SNMF` are in §F.

Fig. 4 illustrates the resulting hierarchy of concepts corresponding to the MLP features learned at each step. Concept labels were derived from the top activating contexts of the MLP feature according to the coefficient matrix. For example, the *weekend* node was activated for contexts mentioning Saturday, Sunday and the word "weekend". Overall, we see that the resulting structure progresses from groups of neurons that represent fine-grained concepts (i.e., specific weekdays) to more high-level concepts (middle/end of the week and "day of week"). This feature-merging behavior is the inverse of what has been described as feature splitting in SAEs (Chanin et al., 2024; Bricken et al., 2023), where a single feature in a smaller SAE corresponds to multiple features in a larger SAE. Additional recursive decomposition examples learned from a more general dataset are in §F.

Semantically-Related MLP Features Share Common Neuron Structures The previous analysis suggests that certain groups of neurons contribute to semantically-related MLP features. Here, we look into these shared structures using the matrix Z. We binarize Z with a per-feature threshold to produce $\bar{Z}$, where the top activating neurons of each MLP feature are set to 1 and the rest to 0. Then, we calculate $M := \bar{Z}\bar{Z}^T$, where each entry $M_{i,j}$ is the number of overlapping, top activating neurons between the MLP features i and j.

Fig. 5 visualizes M, where the first seven rows correspond to MLP features associated with Monday through Sunday and the rest MLP features are randomly sampled from Z. The heatmap reveals that the model represents the concept of weekdays using a common set of neurons, as indicated by the higher shared neuron counts among the first seven rows and columns. Given the increased diagonal values, we understand that each day also activates its own additional distinct subset of neurons. Furthermore, the MLP features representing the weekend and those representing weekdays each share more neurons within their respective groups. We can see this in rows 5 and 6, which have higher shared neuron counts with each other

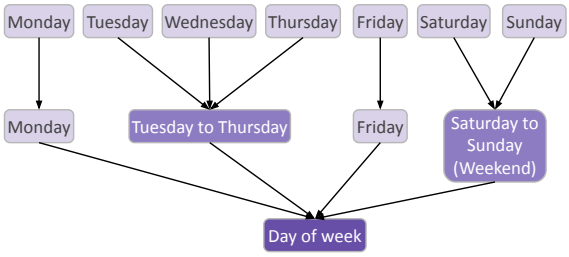

Figure 4: Example feature merging in GPT-2 Small via recursive SNMF, showing the concepts corresponding to the features identified at each step.

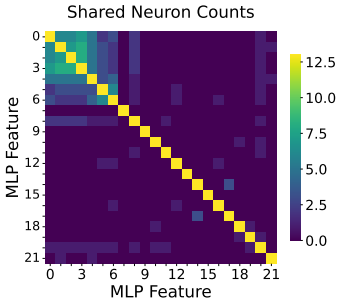

Figure 5: Shared neuron counts between MLP feature pairs. Features 0–6 correspond to weekdays (Monday-Sunday), the rest are randomly sampled from Z. All days share core neurons, suggesting a general day representation, while weekdays and weekends share distinct subsets, forming specialized neuron cores.

than in rows 0 to 4 (and vice versa). Meanwhile, there is little overlap with the remaining MLP features. These patterns show that the model utilizes sets of neurons as building blocks to represent complex and hierarchical concepts. Specifically, a core set of neurons encodes the concept of a day, while additional neurons refine the representation into weekend or weekday, and further neurons produce representations for individual days.

We test this via causal interventions. Let the *neuron base* be the neurons shared across all weekday MLP features, and the *exclusive neurons* the unique neurons to each MLP feature. We feed the prompt "`I think that`" to GPT-2 Large, while amplifying the neuron base and, separately, the exclusive neurons of each weekday. We measure the effect of each intervention on the model's output distribution by calculating the change in logits for tokens associated with each day of the week.

Table 2 shows that amplifying the neuron base promotes all weekday tokens, suggesting it encodes a high-level weekday concept. Conversely, amplifying a day's exclusive neurons promotes its associated token while suppressing the others. For example, activating Sunday's exclusive neurons

Neuron group	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
Monday	**2.0**	-0.8	-1.0	-1.2	-0.8	-1.1	-0.1
Tuesday	-3.3	**0.5**	-2.2	-2.4	-2.4	-3.2	-2.8
Wednesday	-2.7	-3.2	**2.5**	-2.7	-4.3	-3.2	-2.7
Thursday	-0.9	-1.0	**0.6**	**4.5**	-0.1	-1.7	-0.5
Friday	-2.9	-2.8	-2.7	-1.6	**1.3**	-1.9	-2.6
Saturday	-0.2	-0.7	-0.1	-0.7	-0.5	**2.4**	**0.7**
Sunday	-0.4	-1.7	-1.6	-2.0	-0.8	-0.8	**2.6**
Core weekday	**5.8**	**5.7**	**5.4**	**5.8**	**6.0**	**5.4**	**4.7**

Table 2: Change in logits for weekday tokens (columns) when amplifying exclusive neurons tied to different weekday concepts (rows). Positive values indicate promotion; negative indicate suppression. Exclusive neurons promote their own token (diagonal) and suppress others, while core neurons (bottom row) raise all logits, encoding a general "weekday" concept.

suppresses the tokens for "Monday" through "Saturday". These findings demonstrate that the model builds increasingly specific concepts by using sets of neurons as compositional building blocks.

In §G, we quantitatively evaluate hierarchies in MLP layers, showing that broader parent features often cover the causal effects of their children.

7 Conclusion

We introduce an unsupervised method to decompose MLP activations into MLP features, which represent how groups of neurons contribute to encoding concepts. The MLP features often align with human-interpretable concepts, performing on par with SAEs on interpretability benchmarks, while achieving better steering performance than both SAEs and a strong supervised baseline, `DiffMeans`. We further demonstrate that the model composes groups of neurons additively to represent a wide range of concepts, revealing a hierarchical organization in the hidden activation space. This behavior helps explain why the same sets of neurons contribute to multiple features, extending the functional role of neurons in the MLP. It also supports the view that the feature splitting observed in SAEs reflects the model's use of neuron composition to construct more specific features, rather than necessarily being a byproduct of SAE training. Overall, our findings provide a new lens on how MLPs form features in LLMs and we introduce an interpretable unsupervised approach for identifying and interpreting MLP features in transformers.

Limitations

While our work introduces a promising method for analyzing MLP activations, our benchmarks were limited to a feature count of $k < 500$. Although previous research (Mairal et al., 2010) has explored ways to scale NMF in different settings, we focused this paper on establishing the viability of SNMF for activation decomposition. Scaling the method to larger k values requires additional engineering efforts, which we left out of scope to focus on introducing the tool and presenting new findings on concept hierarchies. Exploring the finer-grained features that may emerge at higher k and testing whether the method scales effectively to thousands of features are natural directions for future work.

For evaluation, we rely on LLMs as judges. To increase confidence that using an LLM judge is appropriate in our setting, we follow standard LLM as a judge practices from prior work and additionally verify agreement with humans on our task. Specifically, we ran a human evaluation on a balanced, randomly sampled set of 50 judged examples and measured alignment with the LLM judge, finding strong agreement (Spearman $\rho = 0.8$, $p < 10^{-4}$). While this check supports that the judge behaves consistently with human judgments in our context, results may still exhibit some sensitivity to prompt design and other evaluation details.

Finally, following the original SNMF and NMF papers we use Multiplicative Updates (MU) for the optimization. However, MU can be limiting when extending the method. For future work it may be beneficial to utilize projected gradient descent, which may offer a stronger foundation by supporting regularization and potentially improving performance.

Acknowledgments

We thank Aaron Mueller for his valuable feedback and guidance on training sparse autoencoders. This work was supported in part by the Gemma 2 Academic Research Program at Google, a grant from Open Philanthropy, the Alon Scholarship, and the Israel Science Foundation grant 1083/24.

References

Steven Bills, Nick Cammarata, Dan Mossing, Henk Tillman, Leo Gao, Gabriel Goh, Ilya Sutskever, Jan Leike, Jeff Wu, and William Saunders. 2023. Language models can explain neurons in language models. https://openaipublic.blob.core.windows.net/neuron-explainer/paper/index.html.

Tolga Bolukbasi, Adam Pearce, Ann Yuan, Andy Coenen, Emily Reif, Fernanda Viégas, and Martin Wattenberg. 2021. An interpretability illusion for bert. *arXiv preprint arXiv:2104.07143*.

C. Boutsidis and E. Gallopoulos. 2008. Svd based initialization: A head start for nonnegative matrix factorization. *Pattern Recognition*, 41(4):1350–1362.

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

Tue Minh Cao, Nhat Hoang-Xuan, Hieu Pham, Phi Le Nguyen, and My T. Thai. 2025. Neurflow: Interpreting neural networks through neuron groups and functional interactions. In *The Thirteenth International Conference on Learning Representations*.

David Chanin, James Wilken-Smith, Tomáš Dulka, Hardik Bhatnagar, and Joseph Isaac Bloom. 2024. A is for absorption: Studying feature splitting and absorption in sparse autoencoders. In *Interpretable AI: Past, Present and Future*.

Maheep Chaudhary and Atticus Geiger. 2024. Evaluating open-source sparse autoencoders on disentangling factual knowledge in gpt-2 small. *arXiv preprint arXiv:2409.04478*.

Dami Choi, Vincent Huang, Kevin Meng, Daniel D Johnson, Jacob Steinhardt, and Sarah Schwettmann. 2024. Scaling automatic neuron description. https://transluce.org/neuron-descriptions.

Róbert Csordás, Christopher Potts, Christopher D Manning, and Atticus Geiger. 2024. Recurrent neural networks learn to store and generate sequences using non-linear representations. In *Proceedings of the 7th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 248–262, Miami, Florida, US. Association for Computational Linguistics.

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

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. 2022. Knowledge neurons in pretrained transformers. In *Proceedings of the 60th Annual Meeting of the Association for Computational*

Linguistics (Volume 1: Long Papers), pages 8493–8502, Dublin, Ireland. Association for Computational Linguistics.

Fahim Dalvi, Nadir Durrani, Hassan Sajjad, Yonatan Belinkov, Anthony Bau, and James R. Glass. 2018. What is one grain of sand in the desert? analyzing individual neurons in deep NLP models. *CoRR*, abs/1812.09355.

Chris Ding, Tao Li, and Michael Jordan. 2010. Convex and semi-nonnegative matrix factorizations. ieee trans pattern anal mach intell. *IEEE transactions on pattern analysis and machine intelligence*, 32:45–55.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. 2021. A mathematical framework for transformer circuits. *Transformer Circuits Thread*. Https://transformer-circuits.pub/2021/framework/index.html.

Joshua Engels, Eric J Michaud, Isaac Liao, Wes Gurnee, and Max Tegmark. 2025. Not all language model features are one-dimensionally linear. In *The Thirteenth International Conference on Learning Representations*.

Thomas Fel, Agustin Picard, Louis Béthune, Thibaut Boissin, David Vigouroux, Julien Colin, Rémi Cadène, and Thomas Serre. 2023. Craft: Concept recursive activation factorization for explainability. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2711–2721.

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

Atticus Geiger, Duligur Ibeling, Amir Zur, Maheep Chaudhary, Sonakshi Chauhan, Jing Huang, Aryaman Arora, Zhengxuan Wu, Noah Goodman, Christopher Potts, and Thomas Icard. 2024. Causal abstraction: A theoretical foundation for mechanistic interpretability. *Preprint*, arXiv:2301.04709.

Atticus Geiger, Zhengxuan Wu, Christopher Potts, Thomas Icard, and Noah D. Goodman. 2023. Finding alignments between interpretable causal variables and distributed neural representations. Ms., Stanford University.

Mor Geva, Avi Caciularu, Kevin Wang, and Yoav Goldberg. 2022. Transformer feed-forward layers build predictions by promoting concepts in the vocabulary space. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 30–45, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Google. 2025. Gemini 2.0 Flash. `https://cloud.google.com/vertex-ai/generative-ai/docs/models/gemini/2-0-flash`.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari,

Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Yoav Gur-Arieh, Roy Mayan, Chen Agassy, Atticus Geiger, and Mor Geva. 2025. Enhancing automated interpretability with output-centric feature descriptions. *arXiv preprint arXiv:2501.08319*.

Wes Gurnee, Neel Nanda, Matthew Pauly, Katherine Harvey, Dmitrii Troitskii, and Dimitris Bertsimas. 2023. Finding neurons in a haystack: Case studies with sparse probing. *Transactions on Machine Learning Research*.

Zhengfu He, Wentao Shu, Xuyang Ge, Lingjie Chen, Junxuan Wang, Yunhua Zhou, Frances Liu, Qipeng Guo, Xuanjing Huang, Zuxuan Wu, et al. 2024. Llama scope: Extracting millions of features from llama-3.1-8b with sparse autoencoders. *arXiv preprint arXiv:2410.20526*.

Evan Hernandez, Sarah Schwettmann, David Bau, Teona Bagashvili, Antonio Torralba, and Jacob Andreas. 2021. Natural language descriptions of deep visual features. In *International Conference on Learning Representations*.

Jing Huang, Atticus Geiger, Karel D'Oosterlinck, Zhengxuan Wu, and Christopher Potts. 2023. Rigorously assessing natural language explanations of neurons. In *Proceedings of the 6th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 317–331, Singapore. Association for Computational Linguistics.

Jing Huang, Zhengxuan Wu, Christopher Potts, Mor Geva, and Atticus Geiger. 2024. RAVEL: Evaluating interpretability methods on disentangling language model representations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8669–8687, Bangkok, Thailand. Association for Computational Linguistics.

Fanny Jourdan, Agustin Picard, Thomas Fel, Laurent Risser, Jean-Michel Loubes, and Nicholas Asher. 2023. COCKATIEL: COntinuous concept ranKed ATtribution with interpretable ELements for explaining neural net classifiers on NLP. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 5120–5136, Toronto, Canada. Association for Computational Linguistics.

Andrej Karpathy, Justin Johnson, and Li Fei-Fei. 2015. Visualizing and understanding recurrent networks. *arXiv preprint arXiv:1506.02078*.

Yair Lakretz, German Kruszewski, Theo Desbordes, Dieuwke Hupkes, Stanislas Dehaene, and Marco Baroni. 2019. The emergence of number and syntax units in LSTM language models. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 11–20, Minneapolis, Minnesota. Association for Computational Linguistics.

Daniel Lee and H. Seung. 1999. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788–91.

Justin Lee, Tuomas Oikarinen, Arjun Chatha, Keng-Chi Chang, Yilan Chen, and Tsui-Wei Weng. 2023. The importance of prompt tuning for automated neuron explanations. *Preprint*, arXiv:2310.06200.

Amit Arnold Levy and Mor Geva. 2025. Language models encode numbers using digit representations in base 10. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pages 385–395, Albuquerque, New Mexico. Association for Computational Linguistics.

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

Johnny Lin. 2023. Neuronpedia: Interactive reference and tooling for analyzing neural networks. Software available from neuronpedia.org.

Hanxiao Liu, Zihang Dai, David So, and Quoc V Le. 2021. Pay attention to MLPs. In *Advances in Neural Information Processing Systems*.

Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. 2010. Online learning for matrix factorization and sparse coding. *J. Mach. Learn. Res.*, 11:19–60.

Samuel Marks, Can Rager, Eric J Michaud, Yonatan Belinkov, David Bau, and Aaron Mueller. 2025. Sparse feature circuits: Discovering and editing interpretable causal graphs in language models. In *The Thirteenth International Conference on Learning Representations*.

Samuel Marks and Max Tegmark. 2024. The geometry of truth: Emergent linear structure in large language model representations of true/false datasets. In *First Conference on Language Modeling*.

Luke Melas-Kyriazi, Christian Rupprecht, Iro Laina, and Andrea Vedaldi. 2022. Deep spectral methods: A surprisingly strong baseline for unsupervised semantic segmentation and localization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8364–8375.

Kevin Meng, David Bau, Alex J Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in GPT. In *Advances in Neural Information Processing Systems*.

Abhinav Menon, Manish Shrivastava, David Krueger, and Ekdeep Singh Lubana. 2025. Analyzing (in)abilities of SAEs via formal languages. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4837–4862, Albuquerque, New Mexico. Association for Computational Linguistics.

Jesse Mu and Jacob Andreas. 2020. Compositional explanations of neurons. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA. Curran Associates Inc.

Aaron Mueller, Jannik Brinkmann, Millicent Li, Samuel Marks, Koyena Pal, Nikhil Prakash, Can Rager, Aruna Sankaranarayanan, Arnab Sen Sharma, Jiuding Sun, Eric Todd, David Bau, and Yonatan Belinkov. 2024. The quest for the right mediator: A history, survey, and theoretical grounding of causal interpretability. *Preprint*, arXiv:2408.01416.

Aaron Mueller, Atticus Geiger, Sarah Wiegreffe, Dana Arad, Iván Arcuschin, Adam Belfki, Yik Siu Chan, Jaden Fiotto-Kaufman, Tal Haklay, Michael Hanna, Jing Huang, Rohan Gupta, Yaniv Nikankin, Hadas Orgad, Nikhil Prakash, Anja Reusch, Aruna Sankaranarayanan, Shun Shao, Alessandro Stolfo, Martin Tutek, Amir Zur, David Bau, and Yonatan Belinkov. 2025. Mib: A mechanistic interpretability benchmark. *Preprint*, arXiv:2504.13151.

Neel Nanda, Andrew Lee, and Martin Wattenberg. 2023. Emergent linear representations in world models of self-supervised sequence models. In *Proceedings of the 6th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP, BlackboxNLP@EMNLP 2023, Singapore, December 7, 2023*, pages 16–30. Association for Computational Linguistics.

Tuomas P. Oikarinen and Tsui-Wei Weng. 2024. Linear explanations for individual neurons. In *ICML*.

Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. 2020. Zoom in: An introduction to circuits. *Distill*. Https://distill.pub/2020/circuits/zoom-in.

James Oldfield, Christos Tzelepis, Yannis Panagakis, Mihalis Nicolaou, and Ioannis Patras. 2023. Panda: Unsupervised learning of parts and appearances in the feature maps of GANs. In *The Eleventh International Conference on Learning Representations*.

OpenAI, :, Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, Aleksander Mądry, Alex Baker-Whitcomb, Alex Beutel, Alex Borzunov, Alex Carney, Alex Chow, Alex Kirillov, Alex Nichol, Alex Paino, Alex Renzin, Alex Tachard Passos, Alexander Kirillov, Alexi Christakis, Alexis Conneau, Ali Kamali, Allan Jabri, Allison Moyer, Allison Tam, Amadou Crookes, Amin Tootoochian, Amin Tootoonchian, Ananya Kumar, Andrea Vallone, Andrej Karpathy, Andrew Braunstein, Andrew Cann, Andrew Codispoti, Andrew Galu, Andrew Kondrich, Andrew Tulloch, Andrey Mishchenko, Angela Baek, Angela Jiang, Antoine Pelisse, Antonia Woodford, Anuj Gosalia, Arka Dhar, Ashley Pantuliano, Avi Nayak, Avital Oliver, Barret Zoph, Behrooz Ghorbani, Ben Leimberger, Ben Rossen, Ben Sokolowsky, Ben Wang, Benjamin Zweig, Beth Hoover, Blake Samic, Bob McGrew, Bobby Spero, Bogo Giertler, Bowen Cheng, Brad Lightcap, Brandon Walkin, Brendan Quinn, Brian Guarraci, Brian Hsu, Bright Kellogg, Brydon Eastman, Camillo Lugaresi, Carroll Wainwright, Cary Bassin, Cary Hudson, Casey Chu, Chad Nelson, Chak Li, Chan Jun Shern, Channing Conger, Charlotte Barette, Chelsea Voss, Chen Ding, Cheng Lu, Chong Zhang, Chris Beaumont, Chris Hallacy, Chris Koch, Christian Gibson, Christina Kim, Christine Choi, Christine McLeavey, Christopher Hesse, Claudia Fischer, Clemens Winter, Coley Czarnecki, Colin Jarvis, Colin Wei, Constantin Koumouzelis, Dane Sherburn, Daniel Kappler, Daniel Levin, Daniel Levy, David Carr, David Farhi, David Mely, David Robinson, David Sasaki, Denny Jin, Dev Valladares, Dimitris Tsipras, Doug Li, Duc Phong Nguyen, Duncan Findlay, Edede Oiwoh, Edmund Wong, Ehsan Asdar, Elizabeth Proehl, Elizabeth Yang, Eric Antonow, Eric Kramer, Eric Peterson, Eric Sigler, Eric Wallace, Eugene Brevdo, Evan Mays, Farzad Khorasani, Felipe Petroski Such, Filippo Raso, Francis Zhang, Fred von Lohmann, Freddie Sulit, Gabriel Goh, Gene Oden, Geoff Salmon, Giulio Starace, Greg Brockman, Hadi Salman, Haiming Bao, Haitang Hu, Hannah Wong, Haoyu Wang, Heather Schmidt, Heather Whitney, Heewoo Jun, Hendrik Kirchner, Henrique Ponde de Oliveira Pinto, Hongyu Ren, Huiwen Chang, Hyung Won Chung, Ian Kivlichan, Ian O'Connell, Ian O'Connell, Ian Osband, Ian Silber, Ian Sohl, Ibrahim Okuyucu, Ikai Lan, Ilya Kostrikov, Ilya Sutskever, Ingmar Kanitscheider, Ishaan Gulrajani, Jacob Coxon, Jacob Menick, Jakub Pachocki, James Aung, James Betker, James Crooks, James Lennon, Jamie Kiros, Jan Leike, Jane Park, Jason Kwon, Jason Phang, Jason Teplitz, Jason Wei, Jason Wolfe, Jay Chen, Jeff Harris, Jenia Varavva, Jessica Gan Lee, Jessica Shieh, Ji Lin, Jiahui Yu, Jiayi Weng, Jie Tang, Jieqi Yu, Joanne Jang, Joaquin Quinonero Candela, Joe Beutler, Joe Landers, Joel Parish, Johannes Heidecke, John Schulman, Jonathan Lachman, Jonathan McKay, Jonathan Uesato, Jonathan Ward, Jong Wook Kim, Joost Huizinga, Jordan Sitkin, Jos Kraaijeveld, Josh Gross, Josh Kaplan, Josh Snyder, Joshua Achiam, Joy Jiao, Joyce Lee, Juntang Zhuang, Justyn Harriman, Kai Fricke, Kai Hayashi, Karan Singhal, Katy Shi, Kavin Karthik, Kayla Wood, Kendra Rimbach, Kenny Hsu, Kenny Nguyen, Keren Gu-Lemberg, Kevin Button, Kevin Liu, Kiel Howe, Krithika Muthukumar, Kyle Luther, Lama Ahmad, Larry Kai, Lauren Itow, Lauren Workman, Leher Pathak, Leo Chen, Li Jing, Lia Guy, Liam Fedus, Liang Zhou, Lien Mamitsuka, Lilian Weng, Lindsay McCallum, Lindsey Held, Long Ouyang, Louis Feuvrier, Lu Zhang, Lukas Kondraciuk, Lukasz Kaiser, Luke Hewitt, Luke Metz, Lyric Doshi, Mada Aflak, Maddie Simens, Madelaine Boyd, Madeleine Thompson, Marat Dukhan, Mark Chen, Mark Gray, Mark Hudnall, Marvin Zhang, Marwan Aljubeh, Mateusz Litwin, Matthew Zeng, Max Johnson, Maya Shetty, Mayank Gupta, Meghan Shah, Mehmet Yatbaz, Meng Jia Yang, Mengchao Zhong, Mia Glaese, Mianna Chen, Michael Janner, Michael Lampe, Michael Petrov, Michael Wu,

Michele Wang, Michelle Fradin, Michelle Pokrass, Miguel Castro, Miguel Oom Temudo de Castro, Mikhail Pavlov, Miles Brundage, Miles Wang, Minal Khan, Mira Murati, Mo Bavarian, Molly Lin, Murat Yesildal, Nacho Soto, Natalia Gimelshein, Natalie Cone, Natalie Staudacher, Natalie Summers, Natan LaFontaine, Neil Chowdhury, Nick Ryder, Nick Stathas, Nick Turley, Nik Tezak, Niko Felix, Nithanth Kudige, Nitish Keskar, Noah Deutsch, Noel Bundick, Nora Puckett, Ofir Nachum, Ola Okelola, Oleg Boiko, Oleg Murk, Oliver Jaffe, Olivia Watkins, Olivier Godement, Owen Campbell-Moore, Patrick Chao, Paul McMillan, Pavel Belov, Peng Su, Peter Bak, Peter Bakkum, Peter Deng, Peter Dolan, Peter Hoeschele, Peter Welinder, Phil Tillet, Philip Pronin, Philippe Tillet, Prafulla Dhariwal, Qiming Yuan, Rachel Dias, Rachel Lim, Rahul Arora, Rajan Troll, Randall Lin, Rapha Gontijo Lopes, Raul Puri, Reah Miyara, Reimar Leike, Renaud Gaubert, Reza Zamani, Ricky Wang, Rob Donnelly, Rob Honsby, Rocky Smith, Rohan Sahai, Rohit Ramchandani, Romain Huet, Rory Carmichael, Rowan Zellers, Roy Chen, Ruby Chen, Ruslan Nigmatullin, Ryan Cheu, Saachi Jain, Sam Altman, Sam Schoenholz, Sam Toizer, Samuel Miserendino, Sandhini Agarwal, Sara Culver, Scott Ethersmith, Scott Gray, Sean Grove, Sean Metzger, Shamez Hermani, Shantanu Jain, Shengjia Zhao, Sherwin Wu, Shino Jomoto, Shirong Wu, Shuaiqi, Xia, Sonia Phene, Spencer Papay, Srinivas Narayanan, Steve Coffey, Steve Lee, Stewart Hall, Suchir Balaji, Tal Broda, Tal Stramer, Tao Xu, Tarun Gogineni, Taya Christianson, Ted Sanders, Tejal Patwardhan, Thomas Cunninghman, Thomas Degry, Thomas Dimson, Thomas Raoux, Thomas Shadwell, Tianhao Zheng, Todd Underwood, Todor Markov, Toki Sherbakov, Tom Rubin, Tom Stasi, Tomer Kaftan, Tristan Heywood, Troy Peterson, Tyce Walters, Tyna Eloundou, Valerie Qi, Veit Moeller, Vinnie Monaco, Vishal Kuo, Vlad Fomenko, Wayne Chang, Weiyi Zheng, Wenda Zhou, Wesam Manassra, Will Sheu, Wojciech Zaremba, Yash Patil, Yilei Qian, Yongjik Kim, Youlong Cheng, Yu Zhang, Yuchen He, Yuchen Zhang, Yujia Jin, Yunxing Dai, and Yury Malkov. 2024. Gpt-4o system card. *Preprint*, arXiv:2410.21276.

Kiho Park, Yo Joong Choe, Yibo Jiang, and Victor Veitch. 2025. The geometry of categorical and hierarchical concepts in large language models. In *The Thirteenth International Conference on Learning Representations*.

Kiho Park, Yo Joong Choe, and Victor Veitch. 2024. The linear representation hypothesis and the geometry of large language models. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

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

Gonçalo Paulo and Nora Belrose. 2025. Sparse autoencoders trained on the same data learn different features. *Preprint*, arXiv:2501.16615.

Robert Peharz and Franz Pernkopf. 2012. Sparse nonnegative matrix factorization with ℓ_0-constraints. *Neurocomputing*, 80:38–46. Special Issue on Machine Learning for Signal Processing 2010.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI*. Accessed: 2024-11-15.

Nina Rimsky, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Turner. 2024. Steering llama 2 via contrastive activation addition. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15504–15522, Bangkok, Thailand. Association for Computational Linguistics.

D. E. Rumelhart, J. L. McClelland, and PDP Research Group, editors. 1986. *Parallel Distributed Processing. Volume 1: Foundations*. MIT Press, Cambridge, MA.

Lee Sharkey, Bilal Chughtai, Joshua Batson, Jack Lindsey, Jeff Wu, Lucius Bushnaq, Nicholas Goldowsky-Dill, Stefan Heimersheim, Alejandro Ortega, Joseph Bloom, Stella Biderman, Adria Garriga-Alonso, Arthur Conmy, Neel Nanda, Jessica Rumbelow, Martin Wattenberg, Nandi Schoots, Joseph Miller, Eric J. Michaud, Stephen Casper, Max Tegmark, William Saunders, David Bau, Eric Todd, Atticus Geiger, Mor Geva, Jesse Hoogland, Daniel Murfet, and Tom McGrath. 2025. Open problems in mechanistic interpretability. *Preprint*, arXiv:2501.16496.

Paul Smolensky. 1986. Neural and conceptual interpretation of PDP models. In James L. McClelland, David E. Rumelhart, and the PDP Research Group, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Psychological and Biological Models*, volume 2, pages 390–431. MIT Press.

Hyun Ah Song and Soo-Young Lee. 2013. Hierarchical representation using nmf. In *Neural Information Processing*, pages 466–473, Berlin, Heidelberg. Springer Berlin Heidelberg.

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. 2024. Gemma 2: Improving open language models at a practical size. *Preprint*, arXiv:2408.00118.

Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J. Vazquez, Ulisse Mini, and Monte MacDiarmid. 2024. Steering language models with activation engineering. *Preprint*, arXiv:2308.10248.

Zhengxuan Wu, Aryaman Arora, Atticus Geiger, Zheng Wang, Jing Huang, Dan Jurafsky, Christopher D. Manning, and Christopher Potts. 2025. Axbench: Steering llms? even simple baselines outperform sparse autoencoders. *Preprint*, arXiv:2501.17148.

Zhengxuan Wu, Aryaman Arora, Zheng Wang, Atticus Geiger, Dan Jurafsky, Christopher D Manning, and Christopher Potts. 2024. ReFT: Representation finetuning for language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Zhengxuan Wu, Atticus Geiger, Thomas Icard, Christopher Potts, and Noah Goodman. 2023. Interpretability at scale: Identifying causal mechanisms in alpaca. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Zeyu Yun, Yubei Chen, Bruno Olshausen, and Yann LeCun. 2021. Transformer visualization via dictionary learning: contextualized embedding as a linear superposition of transformer factors. In *Proceedings of Deep Learning Inside Out (DeeLIO): The 2nd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 1–10, Online. Association for Computational Linguistics.

A Activation Decomposition with SNMF

In this section we describe in detail the process used to decompose MLP activations with `SNMF`.

Initialization We compare three initialization strategies for SNMF: **Random** (our baseline), **SVD-based** (Boutsidis and Gallopoulos, 2008), and **K-Means** (Ding et al., 2010). For Random initialization, the non-negative activation matrix $Y \in \mathbb{R}^{k \times n}_{\geq 0}$ is sampled element-wise from $\mathrm{Uniform}[0, 1]$, and the feature matrix $Z \in \mathbb{R}^{d \times k}$ is drawn from $\mathcal{N}(0, 1)$. For SVD and K-Means, we follow the procedures described in their respective references. All variants are trained using multiplicative updates.

Concept Detection. The performance across layers for each initialization is shown below:

Init Strategy	Layer 0	Layer 6	Layer 12	Layer 18	Layer 25	Layer 31
SVD	2.76 ± 1.79	1.63 ± 1.26	0.94 ± 0.92	2.26 ± 1.68	2.07 ± 1.72	0.47 ± 0.88
K-Means	2.55 ± 1.51	1.69 ± 1.32	0.79 ± 0.81	2.45 ± 1.44	1.58 ± 1.58	0.85 ± 0.97
Random	2.99 ± 1.55	1.67 ± 1.39	0.81 ± 0.94	2.35 ± 1.68	1.89 ± 1.72	0.48 ± 0.62

Table 3: Concept detection performance (mean $\pm$ std) across initialization strategies.

Concept Steering and Fluency. We evaluate concept steering quality jointly with generation fluency. Results are consistent across initializations:

Init Strategy	Layer 0	Layer 8	Layer 16	Layer 23	Layer 31
SVD	0.10 ± 0.17	0.28 ± 0.32	0.32 ± 0.29	0.41 ± 0.31	0.25 ± 0.17
K-Means	0.10 ± 0.14	0.29 ± 0.33	0.28 ± 0.28	0.47 ± 0.33	0.30 ± 0.18
Random	0.10 ± 0.12	0.27 ± 0.32	0.31 ± 0.29	0.45 ± 0.32	0.31 ± 0.18

Table 4: Concept steering + fluency scores (mean $\pm$ std) across initialization strategies.

Convergence Speed. While all three methods yield similar performance, K-Means and SVD lead to faster convergence on average:

Init Strategy	Iterations to Convergence
K-Means	1484 ± 1202
SVD	2474 ± 2068
Random	3325 ± 2975

Table 5: Average number of iterations to convergence (mean±std) across all Llama 3.1-8B layers.

Overall, SNMF demonstrates robustness to initialization choice, both in performance and interpretability. The modest convergence benefit of SVD and K-Means makes them practical alternatives, especially in large-scale settings.

Optimization. We also experimented with projected gradient descent during early development and observed performance comparable to multiplicative updates. Given its simplicity and widespread use in SNMF literature, we adopt multiplicative updates in our main experiments. A full optimizer comparison is left for future work.

Feature update With Y fixed we obtain Z:

$$Z \leftarrow AY^\top \big(YY^\top + \lambda I\big)^{-1},$$

where $\lambda > 0$ is a small regularisation constant that prevents ill-conditioning.

Activation update Keeping Z fixed, the non-negative activation matrix Y is updated using the multiplicative rule derived from the SNMF objective:

$$Y \leftarrow Y \odot \sqrt{\frac{\big[Z^\top A\big]_+ + \big[Z^\top Z\big]_- Y}{\big[Z^\top A\big]_- + \big[Z^\top Z\big]_+ Y}},$$

where $[X]_+$ ($[X]_-$) denotes the element-wise positive (negative) part of X, and $\odot$ is the Hadamard product. This update preserves the non-negativity of Y.

Sparsity and Normalization We enforce sparsity in the feature matrix Z using a hard winner-take-all (WTA) operator: at each training step, only the top $p\%$ of entries (by absolute value) are retained per column, with all others zeroed. Unless otherwise stated, we use $p = 1\%$. To stabilize training, each column of the activation matrix Y is normalized to unit ℓ_2 norm, and the corresponding column of Z is rescaled accordingly to keep the product ZY unchanged. This method is adapted from Peharz and Pernkopf (2012), who use a similar approach for sparsity in Non-negative Matrix Factorization.

Sensitivity to Sparsity. We evaluate the effect of varying the WTA threshold ($p \in \{1\%, 5\%, 10\%\}$) on Llama-3.1 8B. As shown below, the $p = 1\%$ setting used in the paper remains optimal across both concept detection and causal steering metrics:

WTA %	Layer 0	Layer 6	Layer 12	Layer 18	Layer 25	Layer 31
1	2.99 ± 1.55	1.67 ± 1.39	0.81 ± 0.94	2.35 ± 1.68	1.89 ± 1.72	0.48 ± 0.62
5	2.60 ± 1.40	1.64 ± 1.19	0.89 ± 0.92	2.29 ± 1.65	1.75 ± 1.46	0.40 ± 0.57
10	2.51 ± 1.56	1.78 ± 1.22	1.03 ± 0.97	1.86 ± 1.62	1.98 ± 1.70	0.51 ± 0.70

Table 6: Concept detection (mean ± std) across different WTA sparsity levels.

WTA %	Layer 0	Layer 8	Layer 16	Layer 23	Layer 31
1	0.10 ± 0.12	0.27 ± 0.32	0.31 ± 0.29	0.45 ± 0.32	0.31 ± 0.18
5	0.07 ± 0.10	0.23 ± 0.27	0.27 ± 0.31	0.41 ± 0.30	0.29 ± 0.18
10	0.05 ± 0.08	0.23 ± 0.27	0.23 ± 0.26	0.34 ± 0.30	0.27 ± 0.18

Table 7: Concept steering + fluency (mean ± std) across different WTA sparsity levels.

The 1% setting provides the best trade-off between sparsity and performance, validating its selection as default in our main experiments.

B Cosine Similarity versus Projection

If we consider the formula for projection we get,

$$\begin{aligned} m_{\text{proj}}(S) &= \frac{1}{|S|} \sum_{\mathbf{x} \in S} \mathbf{f}^\top \mathbf{x} \\ &= \|\mathbf{f}\| \cdot \frac{1}{|S|} \sum_{\mathbf{x} \in S} \|\mathbf{x}\| \cos\theta_x \end{aligned} \tag{7}$$

so for activating S_{act} and neutral S_{neu},

$$\frac{m_{\text{proj}}(S_{\text{act}})}{m_{\text{proj}}(S_{\text{neu}})} = \frac{\sum_{x \in S_{\text{act}}} \|\mathbf{x}\| \cos\theta_x}{\sum_{y \in S_{\text{neu}}} \|\mathbf{y}\| \cos\phi_y} \tag{8}$$

depends on the norms $\|\mathbf{x}\|$, $\|\mathbf{y}\|$. By contrast,

$$m_{\cos}(S) = \frac{1}{|S|} \sum_{\mathbf{x} \in S} \frac{\mathbf{f}^\top \mathbf{x}}{\|\mathbf{f}\|\|\mathbf{x}\|} = \frac{1}{|S|} \sum_{\mathbf{x} \in S} \cos\theta_{\mathbf{x}} \tag{9}$$

cancels $\|\mathbf{f}\|$ and $\|\mathbf{x}\|$. Since both unsupervised methods compare against different activating and neutral sets, using cosine similarity makes the activating/neutral ratio directly comparable across methods.

C Matched-Capacity Baseline SAE Training on MLP Activations

For a matched-capacity baseline in the causal evaluation we trained sparse autoencoders (SAEs) on MLP activations from LLaMA-3.1–8B at layers 0, 8, 16, 23, and 31 and Gemma-2–2B at layers 0, 6, 12, 18, and 25. For concept detection we additionally trained on layers 6, 12, 18, and 25 for LLaMA-3.1–8B. For training, we followed best practices as outlined in Gao et al. (2025). We trained both TopK and L1 regularized architectures across a grid of hyperparameters: three learning rates $(1e{-}4,\ 5e{-}4,\ 1e{-}5)$ and three sparsity levels per architecture. For TopK we used $k \in 3, 5, 10$; for L1-regularized, we used penalties from $(5, 10, 20)$. Each SAE had 100 latents and was trained for 500 epochs. We observed that low regularization tended to produce overly dense features, while increased regularization led to large amounts of dead features. These challenges were particularly difficult to balance with the limited settings, where the smaller number of features (100) and smaller dataset made training less stable and led to training that was more sensitive to hyperparameter choice. To address this, we manually identified a reasonable lower and upper bound for the hyperparameters that didn't create an obvious collapse of training.

We saved 10 checkpoints per run and evaluated them on a held-out validation set constructed identically to the training set, with four sentences per concept. To select the best checkpoint, we combined two metrics: (1) the average number of nonzero activations (to capture sparsity) and (2) downstream loss, measured by the KL divergence between the original model logits and the logits after injecting SAE reconstructions (Gao et al., 2025). We normalized both metrics and combined them to form the selection objective. For Gemma, giving weight to sparsity led to a large amount of dead features, so only downstream loss was used for checkpoint selection. Overall, we trained 252 SAEs and evaluated 2520 checkpoints to identify the SAEs used.

D Comparison of Interpretability Results Across Four K Values

Here we describe the experimental settings of the concept detection benchmarking (§4) and provide comparisons for varying values of k.

D.1 SNMF Dataset

To construct the SNMF training data we randomly sampled 20 concepts from Neuronpedia (Lin, 2023). For each concept we prompted GPT-4o-mini (OpenAI et al., 2024) to generate 10 unique sentences exemplifying the concept. The resulting dataset contains 200 sentences exemplifying 20 labeled concepts and additional non-labeled concepts. For the prompt used to generate the dataset refer to §H.1.

D.2 SNMF Training

We applied SNMF as described in §A with varying values of $k = 100, 200, 300, 400$. For Llama-3.1-8B and Gemma-2-2B, we kept the top 1 percent of indices in each MLP feature during training as described in §A, whereas for GPT-2 we keep the top 5 percent due to its smaller dimensions.

D.3 SAE Concepts

For the SAEs, we randomly sampled 100 features per layer from those identified by Neuronpedia (Lin, 2023), using the SAEs from Llamascope, Gemmascope, and OpenAI's GPT-2 Small (He et al., 2024; Lieberum et al., 2024; Gao et al., 2025), all trained on MLP-out.

D.4 Evaluation

The evaluation procedure is identical across all methods and layers. For each concept and layer, we follow the same methodology: generate 5 activating sentences that exemplify the concept and 5 neutral sentences that do not. Each sentence is passed through the model, and we extract the activations at the target layer. For each activation, we compute the cosine similarity with the feature and record the maximum similarity across all tokens in the sentence. We then average these maximum scores across the activating sentences to obtain $\bar{a}$activating, and across the neutral sentences to obtain $\bar{a}$neutral. Finally, we compute the Concept Detection score (S_{CD}) as the log-ratio between the two averages:

$$S_{CD} := \log \frac{\bar{a}\text{activating}}{\bar{a}\text{neutral}} \tag{10}$$

D.5 Results

Fig. 6, 7 and 8, present results for varying values of k in SNMF, while for SAE the number of concepts is fixed at 100. Across all models, we observe that increasing k generally improves cosine similarity

in most layers, suggesting that larger SNMF decompositions yield features that more cleanly activate for specific concepts. However, there are exceptions. For instance, in Gemma-2-2B at layer 12, performance degrades as k increases. This may indicate that setting k higher than the number of meaningful concepts present at a given layer can hurt performance.

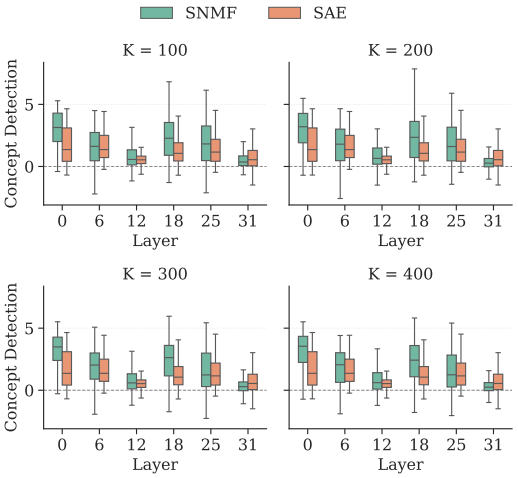

Figure 6: Interpretability scores for Llama-3.1-8B across $K = 100, 200, 300, 400$.

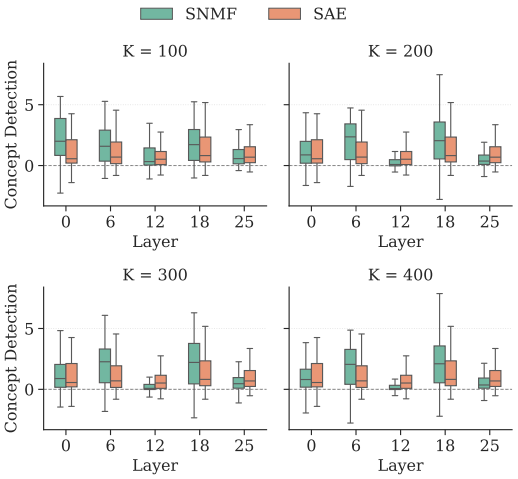

Figure 7: Interpretability scores for Gemma-2-2B across $K = 100, 200, 300, 400$.

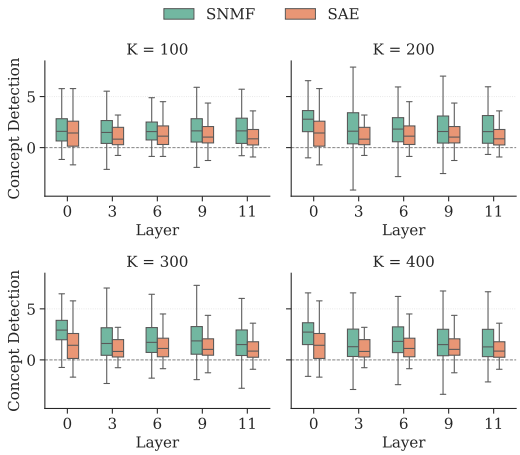

Figure 8: Interpretability scores for GPT-2 Small across $K = 100, 200, 300, 400$.

E Additional Details on Causal Benchmark

E.1 Comparison to Released Causal Benchmarks

We designed the steering evaluation to be aligned with AxBench (Wu et al., 2025): measuring steering success by balancing target concept strength with fluency (instruction following is omitted as we do not evaluate instruction-tuned models). However, AxBench assumes a shared predefined concept set, while unsupervised methods such as SNMF and SAEs generally discover different features rather than the same fixed concepts. We therefore stay as close as possible to their described steering setup, prompts, and scoring, while adapting the concept definition and generation process to maintain a fair symmetric comparison across methods.

E.2 Fluency Scores

Fig. 9 presents the Fluency score results from §5. Notably, we see a low score for Layer 0 showing the increased difficulty in maintaining model coherence at lower layer interventions.

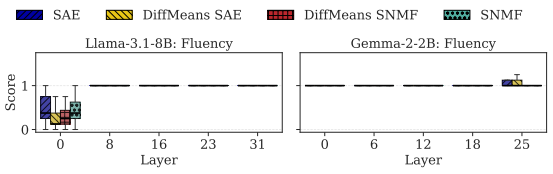

Figure 9: Fluency scores for Llama-3.1-8B and Gemma-2-2B across all methods: SAE, DiffMeans trained on SAE concepts, DiffMeans trained on SNMF concepts and SNMF.

F Experimental Details of Neuron Compositionality

F.1 Recursive SNMF Algorithm

We follow the method outlined in Song and Lee (2013). Initially, we decompose the activations as described in §A to obtain matrices Y_0 and Z_0. Then we recursively decompose the matrix Z_i to produce matrices Y_{i+1} and Z_{i+1}. To improve the integration of the different MLP feature hierarchies, following Song and Lee (2013), we jointly fine-tune all layers using gradient descent to minimize the reconstruction loss between the original activations A and their multi-layer approximation:

$$\mathcal{L} = \frac{1}{2} \|A - Z_L Y_L \cdots Y_1 Y_0\|_F^2$$

The gradients are propagated backward through the factorization layers, and each Z_i and Y_i is updated using gradient descent. This yields a set of multi-level MLP features, where higher-layer features encode increasingly abstract combinations of co-activated neurons.

F.2 Dataset Generation: Time Units

We prompted GPT-o4-mini-high to generate a dataset focused on the concept of units of time: years, months, weekdays, minutes etc. We chose this concept because prior work (Engels et al., 2025) has shown that GPT-2 represents weekdays and months in a multi-dimensional manner, where each day of the week is positioned near the ones that follow it, forming a circular structure in a lower-dimensional subspace. We use the following prompt:

Dataset Generation Prompt

Generate a dataset of sentences that are all related to days of the week, months, years, dates etc. Make sure to include plenty of both broad and specific terms including all names of the days and months and for each term over 10 instances. Generate 200 sentences. Format the dataset with two columns: Prompt, Label. Format as a json in the format <label>: [<sentences>].

F.3 Dataset Generation: General

We constructed a more general dataset by prompting GPT-4o for 50 random nouns as base concepts. For each noun, we queried ConceptNet to retrieve related concepts that represent subcomponents or subcategories using relations like HasA, PartOf, or TypeOf. For each of the related concepts, we prompted Gemini-2.0-flash (Google, 2025) to generate five distinct sentences exemplifying the relationship in natural language. This process resulted in a total of 873 sentences that reflect a diverse range of semantically related sentence groups.

F.4 Tree Construction

To identify hierarchical MLP features, we apply recursive SNMF with $K = [400, 200, 100, 50]$. Each matrix Y_i maps features from level $i{+}1$ to those in Z_i, allowing us to trace how higher-level features activate lower-level ones. To construct the tree, we connect each feature at level $i{+}1$ to its top-activated features in Z_i according to Y_i, using a fixed threshold to prune weak edges. This forms a directed tree structure that reflects the compositional organization of features across levels. We label the nodes corresponding to the MLP features using the top activating contexts associated with the MLP features. We obtain the mapping for an MLP feature at layer i in the hierarchy to the contexts by multiplying the coefficient matrices 0 to i: $Y^{(0\to i)} = Y_0 Y_1 \cdots Y_i$

F.5 Additional Hierarchical Decompositions

In Fig. 10, 11, 12, 13 and 14 we provide additional examples of hierarchical MLP features found by applying the described methodology on the general dataset on MLP activations extracted from Layer 0. We manually select the examples to provide a variety of compositions. We note that these are a select few from a larger pool of identified hierarchical MLP features.

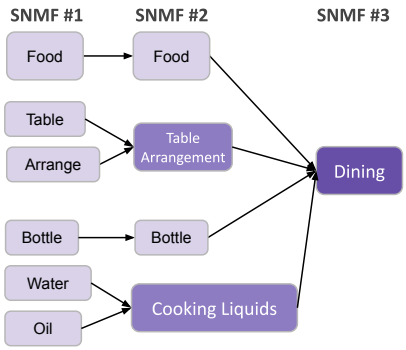

Figure 10: Merging of Dining related MLP features.

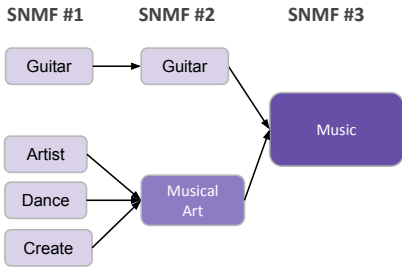

Figure 11: Merging of Music related MLP features.

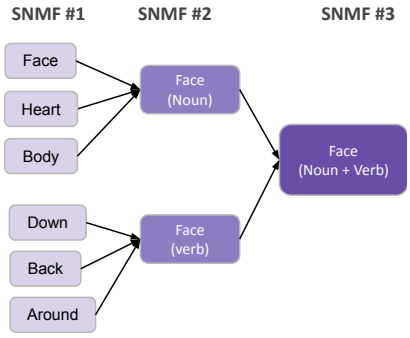

Figure 12: Hierarchy showing that semantically different, but syntactically related features exhibit merging as well.

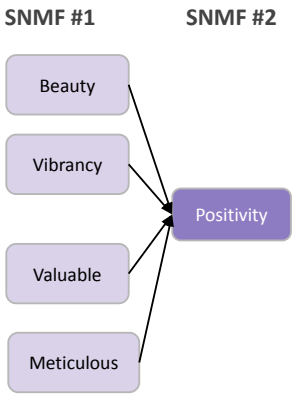

Figure 13: Hierarchy showing that semantically related features share neurons.

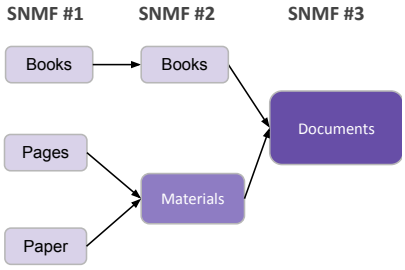

Figure 14: Hierarchy showing that compositional concepts exhibit neuron sharing.

G Hierarchal Features in MLP Layers

To assess whether the identified hierarchy is a property of the MLP, we conducted a broader experiment to evaluate whether a parent feature's causal effect covers the causal effects of its children in their hierarchy. Similar to the phenomenon observed in the weekday example (Table 2).

First, we built a test set with hierarchical concepts. Using GPT-4o, we sampled 50 random "base" nouns, such as "car". For each noun, we queried ConceptNet for HasA, PartOf, SubstanceOf and MemberOf relations, yielding finer-grained sub-concepts (e.g., "a tire" is a part of a car). Then we utilized Gemini-2.0-flash to generate five sentences exemplifying each sub-concept. Overall, we generated a dataset of 873 sentences capturing various hierarchies in language. Next, we obtained activations for these examples and passed them through a two-level recursive SNMF pipeline (ranks K=500, 200) and automatically extracted parent-child links. We consider a lower level feature a child when it contributes at least 2.5% of its parent's column mass which is distributed over 500 features in this setting.

For every parent with at least two children, we measured `coverage` as follows: For a fixed set of 20 initial prompts, we computed the model's logits with and without intervention, intervening with each feature (parent and children) in isolation. For each intervention, we identified the top 50 tokens whose logits increased the most with respect to the non-intervened logits. We then computed `coverage` as the fraction of the children's promoted tokens (the union of their top-10 sets) that also appeared in the parent's top-50 set. Additionally, we measure the `coverage` of each parent with a single child and average across the children in order to validate that the `coverage` is not coming from a single child feature, we term this metric `mean coverage per child`. Together, these metrics quantify how well the parent captures the combined semantic effect of its children.

We report the results for both Gemma-2 2B and Llama-3.1 8B in Table 8. For both models, the average `coverage` was considerably higher than a control baseline, where we randomly selected unrelated features (rather than true parents) and computed their coverage using the same procedure. At the same time, the overlap among children (intersection of children's top 10 promoted logits divided by union of children's top 10 promoted logits)

Metric	Llama-3.1 8B ($n = 93$)		Gemma-2 2B ($n = 67$)	
	Ours	Random	Ours	Random
Coverage	0.46 ± 0.20	0.05 ± 0.09	0.37 ± 0.18	0.00 ± 0.00
Diversity	0.99 ± 0.04	1.00 ± 0.01	0.99 ± 0.03	1.00 ± 0.00
Mean cov/child	0.47 ± 0.20	0.05 ± 0.09	0.38 ± 0.18	0.00 ± 0.00

Table 8: Parent–child feature analysis for two models (mean $\pm$ std). **Coverage**: fraction of tokens promoted by any child feature that are also promoted by its parent. **Diversity**: $1-$overlap among children's top tokens (higher means children specialize in different sub-concepts). **Mean cov/child**: average coverage when each child is considered individually. n = number of parent features evaluated. **Random** = baseline obtained by pairing each parent with the same number of randomly chosen lower-level features.

stayed low, confirming that children specialize in different sub-concepts. Lastly, the mean coverage per child was within 0.02 of the `coverage`, showing the parent is semantically related to each of the children. Since parents consistently cover the union of their children's effects while the children remain diverse, the results provide strong evidence that the broad-to-specific hierarchies we observe are systematic properties of the MLP activation space, not artifacts of the SNMF decomposition.

H Concept Detection Prompts

In this section we present the prompts used for the concept detection experiments.

H.1 Generating Dataset From Concept Descriptions

The following prompt is used sequentially to generate the dataset employed in identifying MLP features with `SNMF` in Section 4 and Section 5. To generate the dataset, we provide the concept and prompt the model m times, where m is the desired number of sentences per concept.

Dataset Generation Prompt

Generate a sentence that exemplifies and strongly exhibits the concept of `{concept}`.
Make sure the sentence is different from the following list: `{current_sentences}`.
Output the sentence **ONLY** without additional text.

H.2 Generating Activating and Neutral Sentences

The following prompts are used to generate sentences that exemplify a concept (activating sentences) and those that do not (neutral sentences). To construct the list, we sequentially prompt the model to generate a new sentence while conditioning on the current list to encourage diversity.

Activating Prompt

You are given a description of an LLM concept.
Given the description, generate a sentence that contains tokens that would activate the feature.
Make sure your generated sentence exemplifies all the key terms and structures specified in the description.
Ensure that the sentence is a full, grammatically correct English sentence.
Category description:
`{description}`
Make sure you generate a sentence that is distinct from: `{previous_sentences}`

Output only the sentence without any additional text.

Neutral Prompt

You are given a description of an LLM concept.
Your objective is to generate a neutral sentence that should not activate the feature.
This means that you must not include in the sentence any tokens that relate to the feature.
Have the generated sentence be on a completely unrelated topic.
Concept description:
`{description}`
Make sure you generate a sentence that is distinct from: `{previous_sentences}`

Output only the sentence without any additional text.
Make sure to exclude any tokens that may activate the concept.

H.3 Input Description Prompt

The following prompt is used to generate a natural language description for each concept, based on the top activated tokens in the matrix Y, which represents token-level activations for each MLP feature. Specifically, for a given MLP feature, we extract the tokens with the top m highest activations and use them as input to the prompt. The model is asked to interpret the shared semantic content of these tokens. We then parse the *Results* section of the model's response to retrieve the final concept description.

Input Description Prompt

You are given a set of tokens, their surrounding context (words before and after the token), and an importance score.
Your task is to determine what is the connection between all the tokens.
Instructions:

1. **Focus on High-Importance Samples:**
   Examine only the token-context pairs with the highest importance scores. If a significant drop is observed beyond a threshold, ignore the lower-scoring pairs.
2. **Assess Token Consistency vs. Contextual Patterns:**
   - **Token Consistency:** If the high-importance samples are mostly identical tokens or strongly related tokens, then consider the tokens only as the primary contributor.
   - **Contextual Patterns:** If the tokens are not related to one another, then focus on common semantic, syntactic, or structural patterns in the surrounding contexts.
3. Always choose the simplest and most obvious underlying connection. If inspecting the tokens alone is enough to find a connection, do not mention or utilize the contexts.

Output Format:

- **Analysis:** `<reason what the underlying connection is>`
- **Results:** `<Single sentence description of the single most obvious connection between the tokens>`

Input:
`Token-Context Pairs: {token_context_str}`
Remember, find the most obvious connection prioritizing connecting the tokens alone without the contexts and only if you cannot find any connection between the tokens then you may inspect their contexts.