CONCEPT COMPONENT ANALYSIS: A PRINCIPLED APPROACH FOR CONCEPT EXTRACTION IN LLMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Extracting human-interpretable concepts from LLMs is essential to understand the mechanisms underlying their capabilities. Sparse autoencoders (SAEs) approach this task by decomposing LLM internal representations into a sparse dictionary, with the hope that the extracted features align with human-interpretable concepts. Despite their empirical progress, SAEs suffer from a fundamental theoretical ambiguity: the relationship between LLM representations and human-interpretable concepts remains unclear. This lack of theoretical grounding gives rise to several methodological challenges, especially how sparsity should be imposed to facilitate concept extraction. In this work, we extend recent theoretical analyses of LLM representations and show that LLM representations can be approximated as a linear mixture of the marginal log-posteriors of individual latent concepts conditioned on the input context. Importantly, this characterization extends beyond the final layer to intermediate layers of LLMs. These findings motivate CONcept Component Analysis (ConCa), a principled formulation of concept extraction as a linear unmixing problem for recovering the log-posterior coordinates of individual concepts. We develop a sparse instantiation of ConCa, termed Sparse ConCa, and demonstrate its ability to extract meaningful concepts across multiple LLMs, offering theory-guided advantages over existing SAEs.

1 INTRODUCTION

One of the critical questions surrounding the practical application of LLMs is the extent to which, and the manner in which, the concepts they encode are grounded in reality. A more specific question is whether a model trained only on natural language can develop representations that capture such grounded concepts (Bowman, 2024; Naveed et al., 2023). Understanding this relationship is crucial, as it informs not only how we interpret model mechanism, but also how we can systematically analyze, evaluate, and manipulate these representations. A promising approach to investigating such questions is to extract human-interpretable concepts, embedded within the models' internal representations and behaviors (Singh et al., 2024). By studying these representations, we can begin to probe which aspects of a model's behavior are aligned with human-interpretable concepts, and how multiple concepts interact to generate model outputs.

1.1 REVISITING SAES FOR CONCEPT EXTRACTION

Sparse autoencoders (SAEs). Recently, SAEs have been explored for this purpose (Elhage et al., 2022; Bricken et al., 2023; Huben et al., 2023), offering a potential perspective through which to analyze model behavior, including how such concepts interact and compose to generate the next token. Technically, SAEs learn a set of features whose *linear* combinations can reconstruct the internal representations of LLMs, while enforcing a *sparsity* prior on the features, in the hope that each feature corresponds to a monosemantic concept (Huben et al., 2023; Gao et al., 2025; Braun et al., 2024; Rajamanoharan et al., 2024a;b; Mudide et al., 2024; Chanin et al., 2024; Lieberum et al., 2024; He et al., 2024; Karvonen et al., 2024; Bussmann et al., 2024).

Hypotheses Behind SAEs. Linearity and sparsity, the two key foundations of SAEs, are jointly expected to promote the emergence of monosemantic and interpretable concepts. The justification for these two foundations primarily relies on two key hypotheses, (i) the linear representation hypothesis

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

and (ii) the superposition hypothesis. The former suggests that concepts are often encoded linearly in LLMs (Tigges et al., 2023; Nanda et al., 2023; Moschella et al., 2022; Park et al., 2023; Li et al., 2024; Gurnee et al., 2023; Rajendran et al., 2024; Jiang et al., 2024), enabling them to be recovered via linear decoding. The latter argues that LLMs tend to represent more features than they have neurons for, leading to overlapping (i.e., superimposed) representations within the same neurons (Elhage et al., 2022). This motivates sparsity as an inductive bias for reducing interference among superimposed features and encouraging more interpretable representations (Huben et al., 2023).

1.2 Motivation and Contributions

While these two hypotheses support SAEs, the deeper theoretical question remains unresolved:

> **Question:** How are human-interpretable concepts actually encoded in LLM representations?

A Deeper Look into SAEs. Without a clear answer, both principled method design and evaluation become major concerns. In terms of method design, for example, while the decoder in SAEs reconstructs representations through *linear* combinations of learned features, the encoder typically includes a *nonlinear* activation function, typically `Relu`, following a linear layer. This architectural asymmetry (i.e., linear decoder *versus* nonlinear encoder) raises a concern about the rationale for introducing the nonlinear activation functions in the encoder. Moreover, without knowing how human-interpretable concepts are represented in LLM activations, it is unclear in which space sparsity should naturally arise. In particular, there is no principled reason to assume that concept representations should be sparse directly in the learned feature space, rather than only after an appropriate transformation of that space. Further, this unclear relationship also makes principled evaluation difficult, i.e., it remains unclear what criteria should be used to determine the quality of the extracted features, as recognized in recent works (Makelov et al., 2024; Kantamneni et al., 2025).

Contributions. We seek to answer this question by building on recent theoretical progress in understanding the relationship between LLM representations and human-interpretable concepts (Park et al., 2023; 2025; Rajendran et al., 2024; Liu et al., 2026). These works formulate concepts as latent variables underlying observed text data, and study how LLM representations learned from text data relate to these latent concepts. In particular, Liu et al. (2026) shows that, under certain conditions, *final-layer* LLM representations can be approximated as a linear mixture of the *joint* log-posteriors over latent concepts conditioned on the input context (Sec. 2). In this work, we extend this result in two respects (Sec. 3): we characterize LLM representations in terms of the *marginal* log-posteriors of latent concepts and extend this characterization across *different layers* of LLMs. These results motivate CONcept Component Analysis (ConCa), a principled formulation of concept extraction as a linear unmixing problem for recovering the marginal log-posteriors of individual concepts (Sec. 4). We develop a sparse instantiation of ConCa, termed Sparse ConCa, and evaluate it across multiple LLM scales and architectures, including Pythia, Gemma3, and Qwen3. Across concept extraction and downstream out-of-distribution tasks, Sparse ConCa consistently outperforms existing SAE variants across a broad range of settings (Sec. 5).

2 Preliminaries: Linear Encoding of Joint Concept Posteriors

The work of Liu et al. (2026) studies the relationship between LLM representations and human-interpretable concepts through a latent-variable generative model of text. In this model, human-interpretable concepts are formulated as discrete latent variables $\mathbf{z}$, while the observed input context $\mathbf{x}$ and next token y are assumed to be generated by an unknown mapping on $\mathbf{z}$. The resulting joint observation distribution is:

$$p(\mathbf{x}, y) = \sum\nolimits_{\mathbf{z}} p(\mathbf{x} \mid \mathbf{z}) p(y \mid \mathbf{z}) p(\mathbf{z}). \tag{1}$$

Here, $\mathbf{z} = (z_1, \ldots, z_K)$ denotes a collection of K discrete latent concepts, where each z_k represents an individual concept and takes values in a concept-specific space $\mathcal{Z}_k$. Their joint space is $\mathcal{Z} = \mathcal{Z}_1 \times \cdots \times \mathcal{Z}_K$, and $\mathbf{z}_i \in \mathcal{Z}$ denotes the i-th possible joint configuration of all K latent concepts.

To examine how information about these latent concepts is encoded in LLM representations, we consider the standard next-token prediction learning framework:

$$p(y \mid \mathbf{x}) = \frac{\exp\left(\mathbf{f}_L(\mathbf{x})^\top \mathbf{g}(y)\right)}{\sum_{y'} \exp\left(\mathbf{f}_L^\top(\mathbf{x})\mathbf{g}(y')\right)}, \tag{2}$$

where $\mathbf{f}_L(\mathbf{x}) \in \mathbb{R}^m$ denotes the final-layer representation of the input context, and $\mathbf{g}(y) \in \mathbb{R}^m$ corresponds to the unembedding vector, associated with token y.

By relating the latent variable model in Eq. (1) to the next-token prediction model in Eq. (2), Liu et al. (2026) establishes that the final-layer representation linearly encodes the joint log-posteriors over the latent concepts, under certain conditions, as follows:

$$\mathbf{f}_L(\mathbf{x}) = \mathbf{A}\left[\log p(\mathbf{z} = \mathbf{z}_i \mid \mathbf{x})\right]_i + \mathbf{b} + \mathbf{e}_L(\mathbf{x}). \tag{3}$$

Here, $\mathbf{A} = (\widehat{\mathbf{L}}^\top)^{-1}\mathbf{L}$, where $\widehat{\mathbf{L}}$ is invertible and $L_{j,i} = p(\mathbf{z} = \mathbf{z}_i \mid y_j) - p(\mathbf{z} = \mathbf{z}_i \mid y_0)$, and $y_0, \ldots, y_m$ are the anchor tokens specified by the diversity condition of (Liu et al., 2026), $[\cdot]_i$ denotes a vector indexed by the $|\mathcal{Z}|$ possible joint configurations, and $\mathbf{e}_L(\mathbf{x})$ denotes the approximation residual. This result characterizes final-layer LLM representations as an approximate linear mixture of the joint log-posteriors. However, even if the mixing effect induced by $\mathbf{A}$ were removed, the resulting coordinates, $[\log p(\mathbf{z} = \mathbf{z}_i \mid \mathbf{x})]_i$, would still correspond to complete joint configurations of $\mathbf{z}$. This characterization does not directly provide a disentangled coordinate system in which each dimension can be associated with a single concept. Moreover, this characterization is restricted to the final-layer representation, leaving unexplained the empirical observation that SAEs could extract concept-related features from intermediate layers (Gao et al., 2024).

3 Concept Posterior Mechanism in LLMs

In this section, we extend the result in Eq. (3) in two directions. First, we move beyond the joint log-posteriors and show that LLM representations can be characterized in terms of the marginal log-posteriors of latent concepts, so that each recovered coordinate can be associated with an individual concept. Second, we generalize it beyond the final layer to intermediate layers, providing theoretical support for why concepts can be extracted from intermediate-layer representations.

To move from the joint log-posteriors in Eq. (3) to the marginal log-posteriors of individual latent concepts, we explicitly separate the joint log-posterior into a component constructed from the marginal log-posteriors and a residual that captures their discrepancy. This decomposition allows us to characterize how posterior interactions enter the LLM representation and, in particular, to isolate their contribution as a representation-level residual. This gives the following:

Corollary 3.1. *Suppose that Eq. (3) holds, and define the log-factorization residual* $r_i(\mathbf{x}) := \log p(\mathbf{z} = \mathbf{z}_i \mid \mathbf{x}) - \sum_{k=1}^{K} \log p(z_k = z_{i,k} \mid \mathbf{x})$, *and let* $\mathbf{r}(\mathbf{x}) := [r_i(\mathbf{x})]_i$. *Then the final-layer representation satisfies*

$$\mathbf{f}_L(\mathbf{x}) = \widetilde{\mathbf{A}}\left[[\log p(z_1 \mid \mathbf{x})]_{z_1}; \ldots; [\log p(z_K \mid \mathbf{x})]_{z_K}\right] + \mathbf{b} + \widetilde{\mathbf{e}}_L(\mathbf{x}), \tag{4}$$

where $\widetilde{\mathbf{A}} \in \mathbb{R}^{m \times \sum_{k=1}^{K} |\mathcal{Z}_k|}$ *is a linear matrix,* $\widetilde{\mathbf{e}}_L(\mathbf{x}) = \mathbf{A}\mathbf{r}(\mathbf{x}) + \mathbf{e}_L(\mathbf{x})$. *Moreover, define* $\gamma := \max_j \mathrm{TV}\left(p(\mathbf{z} \mid y_j), p(\mathbf{z} \mid y_0)\right)$, *where* $y_0, \ldots, y_m$ *are the anchor tokens used to define* $\mathbf{L}$. *Then the interaction residual satisfies* $\|\mathbf{A}\mathbf{r}(\mathbf{x})\|_2 \le 2\sqrt{m}\,\|(\widehat{\mathbf{L}}^\top)^{-1}\|_2 \gamma \max_i |r_i(\mathbf{x})|$.

Proof. See Appendix Sec. C. □

Bound Analysis. Compared with the joint log-posterior characterization in Eq. (3), Eq. (4) decomposes the representation into a component determined by the marginal log-posteriors $[[\log p(z_1 \mid \mathbf{x})]_{z_1}; \ldots; [\log p(z_K \mid \mathbf{x})]_{z_K}]$ and an interaction residual $\mathbf{A}\mathbf{r}(\mathbf{x})$ induced by posterior dependencies among the latent concepts. The bound, $\|\mathbf{A}\mathbf{r}(\mathbf{x})\|_2 \le 2\sqrt{m}\,\|(\widehat{\mathbf{L}}^\top)^{-1}\|_2 \gamma \max_i |r_i(\mathbf{x})|$, shows that the representation-level effect of the interaction $\mathbf{A}\mathbf{r}(\mathbf{x})$ is jointly controlled by the magnitude of the log-factorization residual $\max_i |r_i(\mathbf{x})|$ and the anchor-induced posterior contrasts γ. Intuitively, even when the joint posterior is not exactly factorized and $\mathbf{r}(\mathbf{x})$ is nonzero, its contribution to the representation can remain small when γ is small. This condition might be plausible because a single token often provides limited evidence about the full latent concept, so the corresponding $p(\mathbf{z}|y)$ remain close in total variation (TV). See Sec. 5.1 for empirical evidence.

Implication. Most importantly, when this interaction residual is small, Eq. (4) allows the coordinates indexed by full joint configurations $\mathbf{z} = (z_1, \ldots, z_K)$ to be approximated, up to a small correction, using coordinates associated with individual concepts. Consequently, the representation can be viewed approximately as a linear mixture of the marginal log-posteriors rather than the joint log-posteriors. This reformulation turns concept extraction into a linear unmixing problem: recovering the marginal log-posterior coordinates of individual concepts from their linear mixture; see Sec. 4.

Next, we further extend Corollary 3.1 beyond the final-layer representations. Let $\mathbf{f}_\ell(\mathbf{x})$ denote the representations at layer ℓ. For LLM architecture considered in this work, we define:

$$\mathbf{f}_{\ell+1}(\mathbf{x}) = \mathbf{f}_\ell(\mathbf{x}) + \Phi_\ell(\mathbf{f}_\ell(\mathbf{x})), \quad \ell = 0, \ldots, L-2, \quad \text{and} \quad \mathbf{f}_L(\mathbf{x}) = \mathrm{Norm}\left(\mathbf{f}_{L-1}(\mathbf{x})\right). \tag{5}$$

Here, Φ_ℓ denotes the residual update applied at layer ℓ. Norm denotes the final output normalization applied before the unembedding layer, e.g., LayerNorm or RMSNorm,

$$\mathrm{Norm}\left(\mathbf{f}_{L-1}(\mathbf{x})\right) = \mathbf{G}\frac{\mathbf{f}_{L-1}(\mathbf{x}) - \mathbf{m}(\mathbf{f}_{L-1}(\mathbf{x}))}{s(\mathbf{f}_{L-1}(\mathbf{x}))} + \boldsymbol{\beta}, \tag{6}$$

where $\mathbf{G}$ is the learned diagonal matrix, $\boldsymbol{\beta}$ is an optional bias term, $\mathbf{m}(\mathbf{f}_{L-1}(\mathbf{x}))$ is the centering term, and $s(\mathbf{f}_{L-1}(\mathbf{x}))$ is the scale statistic. Eq. (6) allows us to relate the final-layer representations to intermediate-layer representations. Because the final-layer representation includes an output normalization, this relation contains both the cumulative residual updates from later layers and an output-normalization correction. Specifically, when combined with a local Lipschitz condition on the residual updates, it yields the following layer-wise result.

Corollary 3.2. *Suppose that Corollary 3.1 holds. Assume that the LLM satisfies Eq. (5) with* $\mathbf{G}$ *invertible, and that, for each* $q = 0, \ldots, L-2$*, there exist a reference representation* $\bar{\mathbf{f}}_q$ *and a constant* L_q *such that* $\|\Phi_q(\mathbf{f}_q(\mathbf{x})) - \Phi_q(\bar{\mathbf{f}}_q)\|_2 \le L_q\|\mathbf{f}_q(\mathbf{x}) - \bar{\mathbf{f}}_q\|_2$ *for all* $\mathbf{x} \in \mathrm{supp}(P_\mathbf{x})$*. Then, for every residual depth* $\ell \in \{0, \ldots, L-1\}$*, we have:*

$$\left\|\mathbf{f}_\ell(\mathbf{x}) - \bar{\mathbf{A}}\left[[\log p(z_1 \mid \mathbf{x})]_{z_1}; \ldots; [\log p(z_K \mid \mathbf{x})]_{z_K}\right] - \mathbf{b}_\ell\right\|_2 \le |\bar{s}|\|\mathbf{G}^{-1}\|_2\|\widetilde{\mathbf{e}}_L(\mathbf{x})\|_2 + \rho + \sum_{q=\ell}^{L-2} L_q R_q. \tag{7}$$

Here, $\bar{\mathbf{A}} := \bar{s}\,\mathbf{G}^{-1}\widetilde{\mathbf{A}}$, $\mathbf{b}_\ell := \bar{\mathbf{m}} + \bar{s}\,\mathbf{G}^{-1}(\mathbf{b} - \boldsymbol{\beta}) + \bar{\boldsymbol{\rho}} - \sum_{q=\ell}^{L-2}\Phi_q(\bar{\mathbf{f}}_q)$, $R_q := \sup_{\mathbf{x}\in\mathrm{supp}(P_\mathbf{x})}\|\mathbf{f}_q(\mathbf{x}) - \bar{\mathbf{f}}_q\|_2$*, and the quantities* $\bar{\mathbf{m}}$ *and* $\bar{s}$ *are fixed reference values for the input-dependent centering and scaling statistics* $\mathbf{m}(\mathbf{f}_{L-1}(\mathbf{x}))$ *and* $s(\mathbf{f}_{L-1}(\mathbf{x}))$ *in Eq. (6), respectively. The vector* $\bar{\boldsymbol{\rho}}$ *is the additional fixed reference used to center the residual input-dependent normalization correction. The normalization-statistics variation is:*

$$\rho := \sup_{\mathbf{x}\in\mathrm{supp}(P_\mathbf{x})}\left\|\mathbf{m}(\mathbf{f}_{L-1}(\mathbf{x})) - \bar{\mathbf{m}} + (s(\mathbf{f}_{L-1}(\mathbf{x})) - \bar{s})\,\mathbf{G}^{-1}\left(\mathbf{f}_L(\mathbf{x}) - \boldsymbol{\beta}\right) - \bar{\boldsymbol{\rho}}\right\|_2. \tag{8}$$

Proof. See Appendix Sec. D. □

Bound analysis. Eq. (7) decomposes the layer-wise approximation error into three terms. The first term, $|\bar{s}|\|\mathbf{G}^{-1}\|_2\|\widetilde{\mathbf{e}}_L(\mathbf{x})\|2$, is the final-layer approximation residual rescaled by fixed normalization factors. The second term, ρ, is the *normalization-statistics variation*, which measures the centered input-dependent variation introduced by inverting the output normalization in Eq. (6). The third term, $\sum_{q=\ell}^{L-2} L_q R_q$, is the *residual-tail contribution*, which captures the contribution of the remaining residual tail after layer ℓ. As ℓ approaches $L-1$, this tail contains fewer residual updates and becomes empty at $\ell = L-1$. Therefore, toward later layers, the bound is increasingly dominated by the scaled final-layer residual and the normalization-statistics variation. *This yields a depth-dependent result: the linear encoding of the marginal log-posteriors in Eq. (7) becomes increasingly accurate toward later layers as the residual-tail contribution decreases.*

4 ConCa: A Principled Approach for Concept Extraction

Grounded in the theoretical analysis above, we formulate concept extraction from LLM representations as a linear unmixing problem. Specifically, Corollaries 3.1 and 3.2 show that LLM representations can be approximately expressed as a linear mixture of $[[\log p(z_1 \mid \mathbf{x})]_{z_1}; \ldots; [\log p(z_K \mid \mathbf{x})]_{z_K}]$.

This suggests a natural target for concept extraction: recovering these marginal log-posterior coordinates from the observed LLM representations. We therefore define *CONcept Component Analysis* (ConCa) as the problem of inverting the linear mixture to recover the latent marginal log-posterior components in an unsupervised manner.

4.1 Challenges in Concept Component Analysis

Recovering the marginal log-posteriors from LLM representations presents two main challenges:

> **Challenges:** ① **Ill-posed problem:** The inversion of Eq. (4) (or Eq. (7)) is inherently ill-posed because there may exist many possible solutions that produce the same LLM representations. Without additional constraints or regularization, the solution space is large and ambiguous, leading to non-unique decompositions. ② **Underdetermined problem and overfitting:** The issue above is particularly challenging when the dimension of the latent concept space exceeds that of the observed representations (i.e., $\sum_{k=1}^{K} |\mathcal{Z}_k| > m$), corresponding to the well-known underdetermined case. In this context, overfitting also becomes more likely, since the excess latent dimensionality provides additional flexibility to fit training-specific patterns that may not generalize to testing data.

To address challenge ①, we impose a sparsity prior, requiring that for each context $\mathbf{x}$, only a small subset of latent concepts $\mathbf{z}$ is activated. We refer to this variant as *sparse ConCa*. This inductive bias is motivated by two considerations: (i) it aligns with the superposition hypothesis, a widely discussed phenomenon in the study of SAEs, and (ii) Sparsity provides an inductive bias that restricts the otherwise non-unique decomposition (Elad & Bruckstein, 2002; Gribonval & Schnass, 2010; Spielman et al., 2012; Arora et al., 2014).

We emphasize that one of the key distinctions between SAEs and ConCa lies in how sparsity should be interpreted and enforced. In SAEs, sparsity is directly imposed on the latent feature space, where values near zero correspond to inactive features. In ConCa, however, the situation is inverted. Specifically, the latent feature space in ConCa corresponds to the marginal log-posteriors. A value of $\log p(z_i|\mathbf{x}) = 0$ corresponds to $p(z_i|\mathbf{x}) = 1$, meaning the concept z_i is *fully active* rather than inactive. Consequently, sparsity in ConCa should be enforced in the exponential form of the latent feature space, i.e., the posteriors $p(z_i|\mathbf{x})$, ensuring that only a small subset of concepts is active.

Despite its advantages, sparsity alone may not fully resolve the challenge of overfitting, particularly in underdetermined settings where the dimensionality of $\mathbf{z}$ is much higher than that of $\mathbf{f}$, as mentioned in challenge ②. While the sparsity prior helps restrict the solution space, it may not guarantee generalizable or semantically meaningful decompositions in practical scenarios, especially under limited data and optimization difficulties. In this context, multiple sparse solutions may fit the observed representation equally well, and some may capture meaningless noise or non-semantic patterns rather than true underlying concepts. As a result, techniques to mitigate overfitting may be both useful and even necessary in real applications.

4.2 Sparse ConCa: Architecture and Training Objective

According to the analysis above, we propose sparse ConCa as follows:

$$\hat{\mathbf{z}} = \mathcal{R}(\mathbf{W}_e\mathbf{f}(\mathbf{x}) + \mathbf{b}_e), \qquad \hat{\mathbf{f}}(\mathbf{x}) = \mathbf{W}_d\hat{\mathbf{z}} + \mathbf{b}_d. \tag{9}$$

This is a typical autoencoder architecture, where $\mathcal{R}(\cdot)$ denotes a general regularization module applied to mitigate overfitting, including but not limited to Dropout, and LayerNorm, as the analysis above to address the challenge ②. $\mathbf{W}_e$ and $\mathbf{W}_d$ are learnable weight matrices of the encoder and decoder, respectively, and $\mathbf{b}_e$, $\mathbf{b}_d$ are the corresponding biases. The vector $\hat{\mathbf{z}}$ corresponds to an estimate of $[\log p(z_i \mid \mathbf{x})]_{z_i}$. Let the set of all learnable parameters be $\Theta = \{\mathbf{W}_e, \mathbf{b}_e, \mathbf{W}_d, \mathbf{b}_d\}$. We train the proposed sparse ConCa by minimizing the following objective with respect to Θ:

$$\min_{\Theta} \quad \mathbb{E}_{\mathbf{x}}\left[||\hat{\mathbf{f}}(\mathbf{x}) - \mathbf{f}(\mathbf{x})||_2^2 + \alpha\mathcal{S}(\mathbf{exp}(\hat{\mathbf{z}}))\right], \tag{10}$$

where we apply the $\exp(\cdot)$ function to the representations $\hat{\mathbf{z}}$ (corresponding to log-posterior in theory), to map them back into the probability domain, where sparse activation patterns can be

meaningfully enforced, as motivated by the analysis above to address challenge ①. $\mathcal{S}(\cdot)$ denote the sparsity regularization, which is applied to encourage sparsity on $\mathbf{exp}(\hat{\mathbf{z}})$. This can be implemented using standard sparsity constraints such as L_1 regularization or structured sparsity variants [1]. The hyperparameter α controls the trade-off between reconstruction fidelity and sparsity, allowing the model to be tuned to the expected degree of sparsity.

The key distinctions between our proposed sparse ConCa framework and SAEs are summarized in Table 1. Importantly, the theoretical characterization in Corollary 3.1 directly motivates two key implications for concept extraction in model design:

> **Insights:** 1): Commonly used nonlinear activations, ReLU in SAEs, may not be necessary for concept extraction. 2): Sparsity may be more naturally imposed on the exponentially transformed feature space, i.e., posterior probability space, rather than directly on the raw log-posterior space.

Aspect	SAEs	ConCa
Theoretical grounding	(i) Linear representation hypothesis (ii) Superposition hypothesis	Corollaries 3.1 and 3.2
Objective	Recover monosemantic features	Recover $\log p(z_i\|\mathbf{x})$
Architecture	Encoder: linear + nonlinear activation Decoder: linear	Encoder: linear + module for overfitting Decoder: linear
Role of sparsity	On feature space	On exp-transformed feature space

Table 1: Comparison of SAEs and ConCa. ConCa provides a principled, theoretically grounded framework for concept extraction, while SAEs are largely motivated by empirical hypotheses.

5 Experiments

We organize the experiments around the main theoretical and methodological claims of this work. We first examine Corollary 3.1 by measuring the discrepancy between the joint log-posteriors and their marginal construction, as well as the resulting representation-level interaction residual. We then study Corollary 3.2 by examining the normalization-statistics variation and residual-tail contribution across layers. Finally, we evaluate our proposed Sparse ConCa and compare it with existing SAEs.

5.1 Verifying Corollaries 3.1 and 3.2

Table 2: Empirical validation on E2E.

Model	Acc. ↑	$\widehat{\delta}_r \downarrow$	$\widehat{\delta}_{Ar} \downarrow$
Pythia-1.4B	0.89 ± 0.02	0.26 ± 0.02	0.09 ± 0.01
Gemma-3-1B	0.91 ± 0.02	0.13 ± 0.02	0.04 ± 0.01
Qwen3-1.7B	0.89 ± 0.01	0.24 ± 0.02	0.07 ± 0.01

Corollary 3.1 shows that the joint log-posteriors can be decomposed into a component constructed from the marginal log-posteriors and a log-factorization residual $\mathbf{r}(\mathbf{x})$, which induces the representation-level interaction residual $\mathbf{Ar}(\mathbf{x})$. We therefore empirically examine the magnitude of these residuals on the E2E dataset (Novikova et al., 2017), which contains natural-language restaurant descriptions paired with multiple concepts. We train a multinomial linear probe on the final-layer representations to estimate the joint posterior over 24 concept configurations, denoted by $\widehat{p}_J$ (see Sec. E for theoretical justification). We then obtain the marginal posteriors by exact marginalization of $\widehat{p}_J$, construct their product $\widehat{p}_M$, and compute the corresponding log-factorization residual $\widehat{\mathbf{r}}(\mathbf{x})$. We report the relative residual $\widehat{\delta}_r := \mathrm{RMS}\,\|\widehat{\mathbf{r}}(\mathbf{x})\|_2 / \mathrm{RMS}\,\|[\log \widehat{p}_J(\mathbf{z}_i \mid \mathbf{x})]_i\|_2$, which measures the magnitude of the log-factorization residual relative to the joint log-posterior signal. Here RMS denotes the root mean square. We further report $\widehat{\delta}_{Ar} := \mathrm{RMS}\,\|\widehat{\mathbf{A}}\widehat{\mathbf{r}}(\mathbf{x})\|_2 / \mathrm{RMS}\,\|\widehat{\mathbf{A}}[\log \widehat{p}_J(\mathbf{z}_i \mid \mathbf{x})]_i\|_2$, which measures the remaining representation-level interaction residual after applying $\widehat{\mathbf{A}}$. See Sec. G for details.

[1] We emphasize that sparsity is just a possible regularization choice, other forms are potentially applicable. For instance, non-negativity (Lee & Seung, 1999; Hoyer, 2004) or bounded-range constraints (Cruces, 2010; Erdogan, 2013), since the learned features correspond to probabilities.

As shown in Table 2, the linear probes achieve approximately 90% test accuracy across all three models, providing a reasonable empirical proxy for the joint posterior. The log-factorization residual is also relatively small, with $\widehat{\delta}_r$ ranging from 0.13 to 0.26, suggesting that the marginal log-posteriors provide a reasonable approximation to the joint log-posteriors. More importantly, after applying the estimated mixing matrix, the representation-level interaction residual further decreases to $\widehat{\delta}_{Ar} = 0.04$–$0.09$. Thus, although posterior interactions remain in the joint log-posterior space, their induced residual is further suppressed in the representation space by $\widehat{\mathbf{A}}$. This behavior is consistent across all three models and provides empirical support for the bound characterized in Corollary 3.1.

Next, we conduct an empirical check of Corollary 3.2, whose bound contains three terms. The first, $|\bar{s}|\|\mathbf{G}^{-1}\|_2\|\widetilde{\mathbf{e}}_L(\mathbf{x})\|_2$, is the scaled final-layer approximation residual inherited from Corollary 3.1; the interaction component of this residual, $\mathbf{Ar}(\mathbf{x})$, has been examined in the experiments above. We therefore focus here on the remaining two terms: the normalization-statistics variation ρ in Eq. (8) and the residual-tail contribution $\sum_{q=\ell}^{L-2} L_q R_q$. For ρ, we use the normalization statistics estimated from the training data as fixed references and measure the relative variation induced by the output normalization on test data. For the residual-tail contribution, we measure an empirical counterpart: at each layer, we compute the cumulative residual difference between the layer representation and the final pre-normalization representation. We center this difference by subtracting its mean estimated from the training data and evaluate its magnitude on the test data relative to that of the centered layer representation. More importantly, we also test the depth-dependent implication of the residual-tail contribution: as it decreases toward later layers, the linear encoding of the marginal log-posteriors should become increasingly accurate. Experiments are conducted on the RAVEL dataset (Huang et al., 2024), using Pythia-1.4B, Gemma-3-1B, and Qwen3-1.7B. See Sec. H for more details.

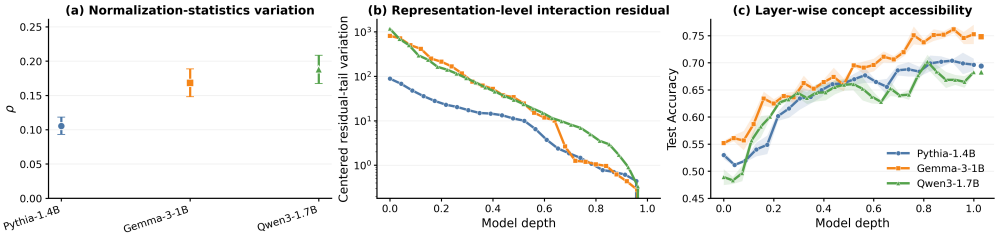

Figure 1: Empirical verification of the layer-wise posterior bound in Corollary 3.2. Left: Normalization-statistics variation induced by the output normalization. Middle: representation-level interaction residual across model depth. Right: Test accuracy of linear probes across layers, showing that concept information becomes increasingly linearly accessible toward later layers.

Fig. 1 shows a consistent pattern across all three model families. First, the normalization-statistics variation remains moderate, with ρ ranging from approximately 0.11 to 0.19. Second, the empirical residual-tail contribution decreases by several orders of magnitude with model depth and vanishes at the final pre-normalization layer. More importantly, linear-probe accuracy increases from approximately 0.49–0.55 in the earliest layers to 0.68–0.75 near the output, while the final normalized readout retains comparable accuracy. Although the accuracy trajectories are not strictly monotonic at every layer, the overall improvement is consistent across Pythia-1.4B, Gemma-3-1B, and Qwen3-1.7B. Together, these results support the implication of Eq. (7): as the residual-tail contribution decreases, the marginal log-posteriors become increasingly accessible through a linear readout.

5.2 Concept Extraction Experiments

We now evaluate the proposed ConCa against several standard SAE methods, including Top-k SAE (Gao et al., 2025), Batch-Top-k SAE (Bussmann et al., 2024), PAnneal SAE (Karvonen et al., 2024), and Structured SAE (Liu et al., 2026), across multiple LLM architectures, including Pythia-70M, 1.4B, and 2.8B (Biderman et al., 2023), Gemma-3-1B (Team et al., 2025), and Qwen3-1.7B (Team, 2025). Training details are provided in Sec. I, and ablation studies of the proposed Sparse ConCa are presented in Sec. L.

We evaluate all methods from two complementary perspectives. First, following SAEBench (Karvonen et al., 2025), we report cosine similarity and mean squared error (MSE) to measure how well

each method reconstructs the original LLM representations. Second, motivated by SAEBench and to align with Corollaries 3.1 and 3.2, we introduce three metrics that evaluate whether the learned representations recover and preserve marginal log-posterior coordinates. See Sec. J for details.

Posterior Alignment measures whether individual learned features align with marginal log-posterior coordinates. Following the evaluation setup of Liu et al. (2026), we construct counterfactual pairs that differ in only one binary concept and train a linear classifier on the corresponding frozen LLM representations (See justification provided in Sec. F based on our results). We use the classifier to estimate the log-posterior for the corresponding binary concept. We compute the Pearson-correlation matrix between these estimated log-posteriors and the learned features, apply a global Hungarian assignment to resolve permutation indeterminacy, and report the mean correlation coefficient (MCC).

Posterior Sensitivity evaluates whether an aligned feature responds selectively to counterfactual changes in its corresponding concept. Using the same counterfactual pairs and feature assignment, we compare the response of the matched feature to changes in the estimated log-posterior of the target concept against its responses to changes in non-target concepts. A larger sensitivity score indicates stronger disentanglement.

Posterior Recovery measures whether marginal log-posterior information is preserved after reconstructing the LLM representation from the learned feature space. For each concept, we apply the same frozen linear probe to both the original and reconstructed representations and report the squared Pearson correlation between the resulting estimated log-posteriors. A larger value indicates better preservation of the corresponding marginal log-posterior information.

Fig. 2 shows that ConCa achieves consistently strong reconstruction performance across all three Pythia models. Its cosine similarity is nearly perfect across model scales, while its MSE is the lowest among all methods. In particular, the improvement in MSE is substantial over Top-k and Batch-Top-k SAEs and remains clear compared with PAnneal and Structured SAE. These results show that the improved posterior alignment of ConCa does not come at the cost of reconstruction fidelity.

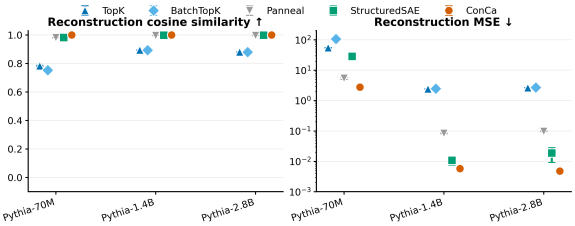

Figure 2: Reconstruction on the `OpenWebText` dataset (Gokaslan & Cohen, 2019).

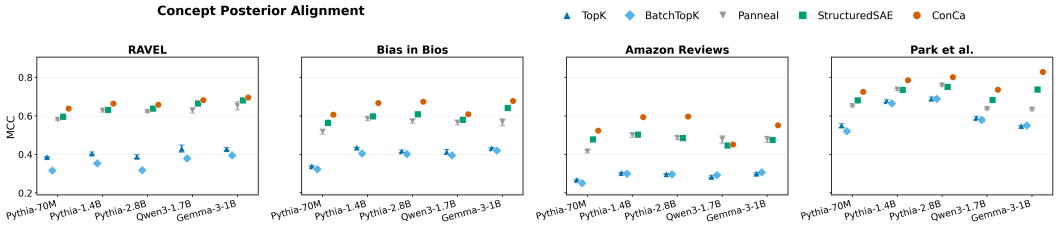

Figure 3: Concept posterior alignment across four datasets and five LLM architectures. The metric evaluates whether individual learned features align with the corresponding marginal log-posteriors.

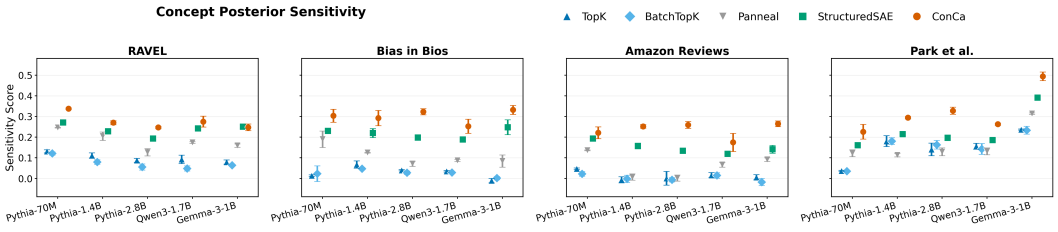

Figure 4: Concept posterior sensitivity across four datasets and five LLM architectures. The metric evaluates whether learned features respond selectively to counterfactual changes in their corresponding concepts while remaining insensitive to non-target concept changes.

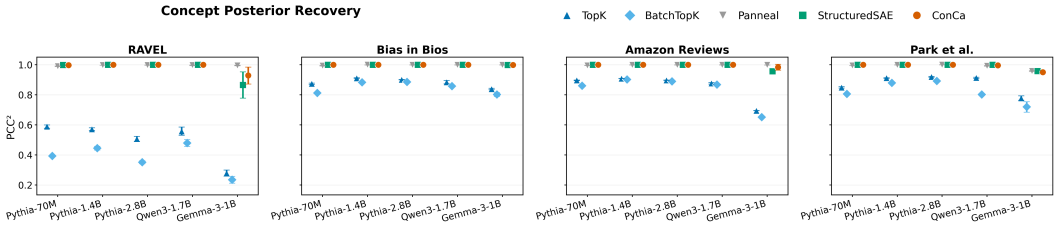

Figure 5: Concept posterior recovery. We report squared Pearson correlation (PCC^2) between marginal log-posterior estimates from the original and reconstructed LLM representations.

Fig. 3 shows that ConCa achieves consistently strong *posterior alignment* across datasets and model architectures. It obtains the highest MCC in nearly all evaluated settings, including across all five LLMs on RAVEL, Bias in Bios, and the counterfactual concepts from Park et al. (2023), while remaining the strongest or highly competitive on Amazon Reviews. Overall, these results show that ConCa more consistently produces individual features aligned with specific marginal log-posterior coordinates. Fig. 4 further evaluates whether these aligned features respond selectively to their corresponding concepts. ConCa achieves the strongest target–off-target separation in nearly all evaluated settings across the four datasets, with particularly clear improvements on Bias in Bios, Amazon Reviews, and the concepts from Park et al. (2023). Since a feature may align with a marginal log-posterior coordinate while still responding to other concepts, the strong sensitivity of ConCa provides complementary evidence that its learned features are more disentangled. Fig. 5 shows a complementary pattern. ConCa, PAnneal, and Structured SAE achieve near-perfect posterior recovery in most settings, whereas Top$k$ and BatchTop$k$ SAEs show noticeably lower squared Pearson correlation (PCC^2) in several cases. This indicates that several methods preserve substantial marginal log-posterior information, but high recovery does not necessarily imply that this information is organized into individual features.

Taken together, the three results distinguish marginal log-posterior information *preservation* from posterior *disentanglement*. While several SAE variants retain substantial marginal log-posterior information, ConCa more consistently combines high recovery with stronger alignment and concept-specific sensitivity, supporting its goal of recovering the marginal log-posteriors.

5.3 Experiments on Out-Of-Distribution Tasks.

In the final experiments, we evaluate out-of-distribution (OOD) generalization to examine how well the features extracted by SAEs and ConCa capture transferable and disentangled concepts. This evaluation is particularly relevant because disentangled representations tend to transfer more robustly under distribution shifts (Fumero et al., 2023). To this end, we evaluate on the OOD tasks from Kantamneni et al. (2025), which include fictional character substitution, random letter insertion, name order reversal, multilingual translation perturbations, and OOD splits from GLUE-X. As shown in Table 3, ConCa achieves the best performance on four of the five LLM backbones and the highest average AUC across backbones. These results indicate that the features learned by ConCa generalize robustly under distribution shifts, providing complementary evidence for their transferability beyond the concept-extraction benchmarks. More details are provided in Sec. K.

Table 3: OOD generalization task across LLMs.

Backbone	TopK	BatchTopK	PAnneal	Struct. SAE	ConCa
Pythia-70M	0.562 ± 0.090	0.563 ± 0.070	$\mathbf{0.624 \pm 0.100}$	0.613 ± 0.107	0.622 ± 0.117
Pythia-1.4B	0.685 ± 0.125	0.664 ± 0.125	0.710 ± 0.117	0.693 ± 0.152	$\mathbf{0.731 \pm 0.116}$
Pythia-2.8B	0.651 ± 0.126	0.687 ± 0.107	0.663 ± 0.145	0.671 ± 0.142	$\mathbf{0.705 \pm 0.111}$
Qwen3-1.7B	0.710 ± 0.145	0.701 ± 0.132	0.688 ± 0.155	0.710 ± 0.190	$\mathbf{0.735 \pm 0.169}$
Gemma-3-1B	0.625 ± 0.104	0.624 ± 0.090	0.612 ± 0.089	0.640 ± 0.105	$\mathbf{0.663 \pm 0.125}$
Average	0.647 ± 0.057	0.648 ± 0.056	0.659 ± 0.042	0.665 ± 0.040	$\mathbf{0.691 \pm 0.048}$

Conclusion Observing the lack of a clear theoretical understanding behind SAEs motivated us to formalize the relationship between LLM representations and human-interpretable concepts. We showed that LLM representations can be approximated as a linear mixture of the marginal log-posteriors, across different layers, which motivates ConCa to recover these concept posteriors in an unsupervised manner. Empirical results across multiple models and benchmarks demonstrate that ConCa extracts features that outperform SAE variants. Looking forward, ConCa opens the door to principled interpretability, analysis, and manipulation of LLM representations.

AI USE STATEMENT

Generative AI tools were used to assist with language editing. All scientific ideas, theoretical results, experimental design, analysis, and conclusions were developed and verified by the authors. The authors take full responsibility for the content of this paper.

ETHICS STATEMENT

This work studies representation learning and concept extraction in large language models using publicly available models and datasets. It does not involve human subjects or private personal data. We are not aware of any specific ethical risks beyond those generally associated with the analysis and interpretation of large language models.

REPRODUCIBILITY STATEMENT

We provide the assumptions and complete proofs of the theoretical results in the appendix. Detailed experimental protocols, dataset processing procedures, model configurations, and evaluation metrics are also provided in the appendix and supplementary materials to facilitate reproduction of our results.

REFERENCES

Michal Aharon, Michael Elad, and Alfred Bruckstein. K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on signal processing*, 54(11): 4311–4322, 2006.

Kartik Ahuja, Divyat Mahajan, Yixin Wang, and Yoshua Bengio. Interventional causal representation learning. In *ICML*, pp. 372–407. PMLR, 2023.

Sanjeev Arora, Rong Ge, and Ankur Moitra. New algorithms for learning incoherent and overcomplete dictionaries. In *Conference on Learning Theory*, pp. 779–806. PMLR, 2014.

Sanjeev Arora, Rong Ge, Tengyu Ma, and Ankur Moitra. Simple, efficient, and neural algorithms for sparse coding. In *Conference on learning theory*, pp. 113–149. PMLR, 2015.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pp. 2397–2430. PMLR, 2023.

Samuel R Bowman. Eight things to know about large language models. *Critical AI*, 2(2), 2024.

Dan Braun, Jordan Taylor, Nicholas Goldowsky-Dill, and Lee Sharkey. Identifying functionally important features with end-to-end sparse dictionary learning. *Advances in Neural Information Processing Systems*, 37:107286–107325, 2024.

Johann Brehmer, Pim De Haan, Phillip Lippe, and Taco Cohen. Weakly supervised causal representation learning. *arXiv preprint arXiv:2203.16437*, 2022.

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

Simon Buchholz, Goutham Rajendran, Elan Rosenfeld, Bryon Aragam, Bernhard Schölkopf, and Pradeep Ravikumar. Learning linear causal representations from interventions under general nonlinear mixing. *arXiv preprint arXiv:2306.02235*, 2023.

Bart Bussmann, Patrick Leask, and Neel Nanda. Batchtopk sparse autoencoders. *arXiv preprint arXiv:2412.06410*, 2024.

Yichao Cai, Yuhang Liu, Erdun Gao, Tianjiao Jiang, Zhen Zhang, Anton van den Hengel, and Javen Qinfeng Shi. On the value of cross-modal misalignment in multimodal representation learning. *arXiv preprint arXiv:2504.10143*, 2025.

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

Sergio Cruces. Bounded component analysis of linear mixtures: A criterion of minimum convex perimeter. *IEEE Transactions on Signal Processing*, 58(4):2141–2154, 2010.

Maria De-Arteaga, Alexey Romanov, Hanna Wallach, Jennifer Chayes, Christian Borgs, Alexandra Chouldechova, Sahin Geyik, Krishnaram Kenthapadi, and Adam Tauman Kalai. Bias in bios: A case study of semantic representation bias in a high-stakes setting. In *proceedings of the Conference on Fairness, Accountability, and Transparency*, pp. 120–128, 2019.

Bogdan Dumitrescu and Paul Irofti. *Dictionary learning algorithms and applications*. Springer, 2018.

Julian Eggert and Edgar Korner. Sparse coding and nmf. In *2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No. 04CH37541)*, volume 4, pp. 2529–2533. IEEE, 2004.

Michael Elad. *Sparse and redundant representations: from theory to applications in signal and image processing*. Springer Science & Business Media, 2010.

Michael Elad and Alfred M Bruckstein. A generalized uncertainty principle and sparse representation in pairs of bases. *IEEE Transactions on Information Theory*, 48(9):2558–2567, 2002.

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

Alper T Erdogan. A class of bounded component analysis algorithms for the separation of both independent and dependent sources. *IEEE Transactions on Signal Processing*, 61(22):5730–5743, 2013.

Marco Fumero, Florian Wenzel, Luca Zancato, Alessandro Achille, Emanuele Rodolà, Stefano Soatto, Bernhard Schölkopf, and Francesco Locatello. Leveraging sparse and shared feature activations for disentangled representation learning. In *Advances in Neural Information Processing Systems*, 2023.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2020.

Leo Gao, Tom Dupré la Tour, Henk Tillman, Gabriel Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan Leike, and Jeffrey Wu. Scaling and evaluating sparse autoencoders. *arXiv preprint arXiv:2406.04093*, 2024.

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

Aaron Gokaslan and Vanya Cohen. Openwebtext corpus. http://Skylion007.github.io/OpenWebTextCorpus, 2019.

Rémi Gribonval and Karin Schnass. Dictionary identification—sparse matrix-factorization via ℓ_1-minimization. *IEEE Transactions on Information Theory*, 56(7):3523–3539, 2010.

Yuqi Gu and David B Dunson. Bayesian pyramids: Identifiable multilayer discrete latent structure models for discrete data. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 85(2):399–426, 2023.

Wes Gurnee, Neel Nanda, Matthew Pauly, Katherine Harvey, Dmitrii Troitskii, and Dimitris Bertsimas. Finding neurons in a haystack: Case studies with sparse probing. *arXiv preprint arXiv:2305.01610*, 2023.

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

Patrik O Hoyer. Non-negative matrix factorization with sparseness constraints. *Journal of machine learning research*, 5(Nov):1457–1469, 2004.

Jing Huang, Zhengxuan Wu, Christopher Potts, Mor Geva, and Atticus Geiger. 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)*, pp. 8669–8687, 2024.

Robert Huben, Hoagy Cunningham, Logan Riggs Smith, Aidan Ewart, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. In *The Twelfth International Conference on Learning Representations*, 2023.

Aapo Hyvarinen and Hiroshi Morioka. Unsupervised feature extraction by time-contrastive learning and nonlinear ica. *Advances in neural information processing systems*, 29, 2016.

Aapo Hyvarinen, Hiroaki Sasaki, and Richard Turner. Nonlinear ica using auxiliary variables and generalized contrastive learning. In *The 22nd international conference on artificial intelligence and statistics*, pp. 859–868. PMLR, 2019.

Yibo Jiang, Goutham Rajendran, Pradeep Ravikumar, Bryon Aragam, and Victor Veitch. On the origins of linear representations in large language models. *arXiv preprint arXiv:2403.03867*, 2024.

Subhash Kantamneni, Joshua Engels, Senthooran Rajamanoharan, Max Tegmark, and Neel Nanda. Are sparse autoencoders useful? a case study in sparse probing. *arXiv preprint arXiv:2502.16681*, 2025.

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

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

Ilyes Khemakhem, Diederik Kingma, Ricardo Monti, and Aapo Hyvarinen. Variational autoencoders and nonlinear ica: A unifying framework. In *International conference on artificial intelligence and statistics*, pp. 2207–2217. PMLR, 2020.

Bohdan Kivva, Goutham Rajendran, Pradeep Ravikumar, and Bryon Aragam. Learning latent causal graphs via mixture oracles. *Advances in Neural Information Processing Systems*, 34:18087–18101, 2021.

Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept bottleneck models. In *International conference on machine learning*, pp. 5338–5348. PMLR, 2020.

Lingjing Kong, Guangyi Chen, Biwei Huang, Eric P Xing, Yuejie Chi, and Kun Zhang. Learning discrete concepts in latent hierarchical models. *arXiv preprint arXiv:2406.00519*, 2024.

Daniel D Lee and H Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *nature*, 401(6755):788–791, 1999.

Tobias Leemann, Michael Kirchhof, Yao Rong, Enkelejda Kasneci, and Gjergji Kasneci. When are post-hoc conceptual explanations identifiable? In *Uncertainty in Artificial Intelligence*, pp. 1207–1218. PMLR, 2023.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Inference-time intervention: Eliciting truthful answers from a language model. *Advances in Neural Information Processing Systems*, 36, 2024.

Tom Lieberum, Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Nicolas Sonnerat, Vikrant Varma, János Kramár, Anca Dragan, Rohin Shah, and Neel Nanda. Gemma scope: Open sparse autoencoders everywhere all at once on gemma 2. *arXiv preprint arXiv:2408.05147*, 2024.

Yuhang Liu, Dong Gong, Yichao Cai, Erdun Gao, Zhen Zhang, Biwei Huang, Mingming Gong, Anton van den Hengel, and Javen Qinfeng Shi. I predict therefore i am: Is next token prediction enough to learn human-interpretable concepts from data? In *The Fourteenth International Conference on Learning Representations*, 2026.

Aleksandar Makelov, George Lange, and Neel Nanda. Towards principled evaluations of sparse autoencoders for interpretability and control. *arXiv preprint arXiv:2405.08366*, 2024.

Emanuele Marconato, Andrea Passerini, and Stefano Teso. Interpretability is in the mind of the beholder: A causal framework for human-interpretable representation learning. *Entropy*, 25(12): 1574, 2023.

Emanuele Marconato, Sébastien Lachapelle, Sebastian Weichwald, and Luigi Gresele. All or none: Identifiable linear properties of next-token predictors in language modeling. *arXiv preprint arXiv:2410.23501*, 2024.

Riccardo Massidda, Atticus Geiger, Thomas Icard, and Davide Bacciu. Causal abstraction with soft interventions. In *Conference on Causal Learning and Reasoning*, pp. 68–87. PMLR, 2023.

Luca Moschella, Valentino Maiorca, Marco Fumero, Antonio Norelli, Francesco Locatello, and Emanuele Rodolà. Relative representations enable zero-shot latent space communication. *arXiv preprint arXiv:2209.15430*, 2022.

Anish Mudide, Joshua Engels, Eric J Michaud, Max Tegmark, and Christian Schroeder de Witt. Efficient dictionary learning with switch sparse autoencoders. *arXiv preprint arXiv:2410.08201*, 2024.

Neel Nanda, Andrew Lee, and Martin Wattenberg. Emergent linear representations in world models of self-supervised sequence models. *arXiv preprint arXiv:2309.00941*, 2023.

Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes, and Ajmal Mian. A comprehensive overview of large language models. *arXiv preprint arXiv:2307.06435*, 2023.

Jianmo Ni, Jiacheng Li, and Julian McAuley. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pp. 188–197, 2019.

Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. The e2e dataset: New challenges for end-to-end generation. In *Proceedings of the 18th annual SIGdial meeting on discourse and dialogue*, pp. 201–206, 2017.

Tuomas Oikarinen, Subhro Das, Lam M Nguyen, and Tsui-Wei Weng. Label-free concept bottleneck models. *arXiv preprint arXiv:2304.06129*, 2023.

Kiho Park, Yo Joong Choe, and Victor Veitch. The linear representation hypothesis and the geometry of large language models. *arXiv preprint arXiv:2311.03658*, 2023.

Kiho Park, Yo Joong Choe, Yibo Jiang, and Victor Veitch. The geometry of categorical and hierarchical concepts in large language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=bVTM2QKYuA.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

Eleonora Poeta, Gabriele Ciravegna, Eliana Pastor, Tania Cerquitelli, and Elena Baralis. Concept-based explainable artificial intelligence: A survey. *arXiv preprint arXiv:2312.12936*, 2023.

Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Tom Lieberum, Vikrant Varma, János Kramár, Rohin Shah, and Neel Nanda. Improving dictionary learning with gated sparse autoencoders. *arXiv preprint arXiv:2404.16014*, 2024a.

Senthooran Rajamanoharan, Tom Lieberum, Nicolas Sonnerat, Arthur Conmy, Vikrant Varma, János Kramár, and Neel Nanda. Jumping ahead: Improving reconstruction fidelity with jumprelu sparse autoencoders. *arXiv preprint arXiv:2407.14435*, 2024b.

Goutham Rajendran, Simon Buchholz, Bryon Aragam, Bernhard Schölkopf, and Pradeep Ravikumar. Learning interpretable concepts: Unifying causal representation learning and foundation models. *arXiv preprint arXiv:2402.09236*, 2024.

Geoffrey Roeder, Luke Metz, and Durk Kingma. On linear identifiability of learned representations. In *International Conference on Machine Learning*, pp. 9030–9039. PMLR, 2021.

Bernhard Schölkopf, Francesco Locatello, Stefan Bauer, Nan Rosemary Ke, Nal Kalchbrenner, Anirudh Goyal, and Yoshua Bengio. Toward causal representation learning. *Proceedings of the IEEE*, 109(5):612–634, 2021.

Lisa Schut, Nenad Tomasev, Tom McGrath, Demis Hassabis, Ulrich Paquet, and Been Kim. Bridging the human-ai knowledge gap: Concept discovery and transfer in alphazero. *arXiv preprint arXiv:2310.16410*, 2023.

Gesina Schwalbe. Concept embedding analysis: A review. *arXiv preprint arXiv:2203.13909*, 2022.

Anna Seigal, Chandler Squires, and Caroline Uhler. Linear causal disentanglement via interventions. *arXiv preprint arXiv:2211.16467*, 2022.

Xinwei Shen, Furui Liu, Hanze Dong, Qing Lian, Zhitang Chen, and Tong Zhang. Weakly supervised disentangled generative causal representation learning. *The Journal of Machine Learning Research*, 23(1):10994–11048, 2022.

Chandan Singh, Jeevana Priya Inala, Michel Galley, Rich Caruana, and Jianfeng Gao. Rethinking interpretability in the era of large language models. *arXiv preprint arXiv:2402.01761*, 2024.

Daniel A Spielman, Huan Wang, and John Wright. Exact recovery of sparsely-used dictionaries. In *Conference on Learning Theory*, pp. 37–1. JMLR Workshop and Conference Proceedings, 2012.

Armeen Taeb, Nicolo Ruggeri, Carina Schnuck, and Fanny Yang. Provable concept learning for interpretable predictions using variational autoencoders. *arXiv preprint arXiv:2204.00492*, 2022.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, et al. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025.

Qwen Team. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Curt Tigges, Oskar John Hollinsworth, Atticus Geiger, and Neel Nanda. Linear representations of sentiment in large language models. *arXiv preprint arXiv:2310.15154*, 2023.

Burak Varici, Emre Acarturk, Karthikeyan Shanmugam, Abhishek Kumar, and Ali Tajer. Score-based causal representation learning with interventions. *arXiv preprint arXiv:2301.08230*, 2023.

Julius Von Kügelgen, Yash Sharma, Luigi Gresele, Wieland Brendel, Bernhard Schölkopf, Michel Besserve, and Francesco Locatello. Self-supervised learning with data augmentations provably isolates content from style. In *NeurIPS*, 2021.

Julius von Kügelgen, Michel Besserve, Liang Wendong, Luigi Gresele, Armin Kekić, Elias Bareinboim, David Blei, and Bernhard Schölkopf. Nonparametric identifiability of causal representations from unknown interventions. *Advances in Neural Information Processing Systems*, 36, 2023.

Yue Yang, Artemis Panagopoulou, Shenghao Zhou, Daniel Jin, Chris Callison-Burch, and Mark Yatskar. Language in a bottle: Language model guided concept bottlenecks for interpretable image classification. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 19187–19197, 2023.

Appendix

Table of Contents

A Related Work 17

B Theoretical Limitations and Justification 18

C Proof of Corollary 3.1 18

D Proof of Corollary 3.2 19

E Joint Posterior Estimation via Linear Probe 21

F Binary Log-Posterior Estimation via Linear Probing 21

G Experimental Details for Verifying Corollary 3.1 22

H Experimental Details for Verifying Corollary 3.2 23

I Training Details for Concept Extraction Experiments 24

J Evaluation Details of Concept Extraction Experiments 25

K Experimental Details for Out-of-Distribution Downstream Tasks 30

L Ablation Studies for Sparse ConCa 30

A Related Work

Sparse Autoencoders and Dictionary Learning The proposed ConCa framework is closely related to Sparse Autoencoders (SAEs) (Rajamanoharan et al., 2024a;b; Gao et al., 2025; Braun et al., 2024; Bricken et al., 2023; Huben et al., 2023; Gao et al., 2025; Mudide et al., 2024; Chanin et al., 2024; Lieberum et al., 2024; He et al., 2024; Karvonen et al., 2024; Bussmann et al., 2024), as both aim to extract and monosemantic human-interpretable concepts from LLM representations in order to provide mechanistic explanations for their success. However, the two approaches differ fundamentally in their theoretical foundations. ConCa is grounded in a rigorous theoretical framework, as established in Corollaries 3.1 and 3.2, while SAEs rely on the linear representation hypothesis and the superposition hypothesis. This foundational difference leads to notable divergences in both method design and evaluation protocols, as discussed in the Introduction. In addition, our work is also closely connected to the well-established framework of dictionary learning (Dumitrescu & Irofti, 2018; Eggert & Korner, 2004; Elad, 2010; Elad & Bruckstein, 2002; Aharon et al., 2006; Arora et al., 2015). Specifically, this work bridges next-token prediction framework and dictionary learning by showing that LLM representations acquired through the next-token prediction framework can be further meaningfully decomposed.

Causal Representation Learning This work is also related to causal representation learning (Schölkopf et al., 2021), which seeks to identify latent causal variables from observational data (Brehmer et al., 2022; Von Kügelgen et al., 2021; Massidda et al., 2023; von Kügelgen et al., 2023; Ahuja et al., 2023; Seigal et al., 2022; Shen et al., 2022; Buchholz et al., 2023; Varici et al., 2023; Hyvarinen & Morioka, 2016; Hyvarinen et al., 2019; Khemakhem et al., 2020; Cai et al., 2025; Rajendran et al., 2024). Most of those works focus on continuous latent and observed variables, we explore the setting of discrete variables. A subset of studies has investigated causal representation learning in discrete spaces (Gu & Dunson, 2023; Kong et al., 2024; Kivva et al., 2021), but these typically assume specific graphical structures and rely on invertible mappings from latent to observed variables. In contrast, our approach does not require such assumptions, offering greater flexibility.

Identifiability Analysis for LLMs Several prior studies (Marconato et al., 2024; Roeder et al., 2021) have explored identifiability within the inference space, revealing alignments between representations obtained from distinct inference models. However, these findings remain confined to the inference space and do not extend to identifying the true latent variables in latent variable models. More recently, Jiang et al. (2024) examined the emergence of linear structures under a different generative framework, attributing them to the implicit bias introduced by gradient-based optimization. In contrast, our approach offers a theoretical explanation rooted in identifiability theory, directly linking the observed linear patterns to the ground-truth latent structure. This shift in perspective provides a deeper and more principled understanding of the underlying mechanisms.

Concept Discovery Concept discovery aims to extract human-interpretable concepts from pre-trained models, and has emerged as a key area within machine learning (Schut et al., 2023; Yang et al., 2023; Marconato et al., 2023; Oikarinen et al., 2023; Koh et al., 2020; Schwalbe, 2022; Poeta et al., 2023; Taeb et al., 2022). While empirical methods have flourished, theoretical understanding of when and how such concepts can be reliably identified remains limited. In contrast, the proposed ConCa is grounded in rigorous theoretical results. The work of Leemann et al. (2023) investigates concept identifiability under the assumption that the non-linear mapping is known a priori. In contrast, our results provide a theoretical characterization without requiring such prior knowledge. A recent advance by Rajendran et al. (2024) offers formal identifiability results for continuous latent concepts under a likelihood-matching framework, while our work focuses on discrete concepts, and approaches the problem from a different angle, i.e., rooted in the next-token prediction paradigm, which underpins modern LLMs training. This shift in both focus and framework allows us to derive new identifiability for the discrete setting.

B Theoretical Limitations and Justification

Our theoretical analysis in Corollaries 3.1 and 3.2 relies on certain assumptions about the true generative model that cannot be directly verified in real-world text. In particular, Corollary 3.1 characterizes the joint log-posteriors in terms of the marginal log-posteriors and an interaction residual $\mathbf{Ar}(\mathbf{x})$, but neither the underlying latent posterior nor the true mixing matrix $\mathbf{A}$ is directly available in practice. Therefore, this characterization cannot be checked exactly. Nevertheless, we can examine it empirically by estimating the joint posterior and the mixing matrix from LLM representations, and measuring both the log-factorization residual and the resulting representation-level interaction residual. The results in Sec. 5.1 are consistent with the characterization in Corollary 3.1.

Second, Corollary 3.2 assumes a local Lipschitz condition on each residual update, $\|\Phi_q(\mathbf{f}_q(\mathbf{x})) - \Phi_q(\bar{\mathbf{f}}_q)\|_2 \leq L_q\|\mathbf{f}_q(\mathbf{x}) - \bar{\mathbf{f}}_q\|_2$. Such local smoothness conditions are commonly used in the analysis of deep neural networks to control how differences propagate through successive layers. Importantly, our result requires this condition only relative to a reference representation over the support of the data. In practice, directly estimating the constants L_q can lead to very loose bounds. We therefore examine an empirical counterpart of the residual-tail contribution. The observed decrease in this quantity with model depth, together with the increasing linear accessibility of the marginal log-posteriors in later layers, is consistent with the depth-dependent behavior characterized by Corollary 3.2. These results provide empirical support for the behavior implied by the bound.

C Proof of Corollary 3.1

Proof. For each concept z_k, define

$$\boldsymbol{\ell}_k(\mathbf{x}) := [\log p(z_k = a \mid \mathbf{x})]_{a \in \mathcal{Z}_k}, \tag{11}$$

and stack these vectors as

$$\boldsymbol{\ell}_{\text{ind}}(\mathbf{x}) := [\boldsymbol{\ell}_1(\mathbf{x}); \ldots; \boldsymbol{\ell}_K(\mathbf{x})] = [[\log p(z_1 \mid \mathbf{x})]_{z_1}; \ldots; [\log p(z_K \mid \mathbf{x})]_{z_K}]. \tag{12}$$

For each joint configuration $\mathbf{z}_i = (z_{i,1}, \ldots, z_{i,K})$, the term

$$\sum_{k=1}^{K} \log p(z_k = z_{i,k} \mid \mathbf{x})$$

selects one coordinate from each marginal log-posterior vector and sums them. Therefore, there exists a fixed binary selection matrix

$$\mathbf{S} \in \mathbb{R}^{|\mathcal{Z}| \times \sum_{k=1}^{K} |\mathcal{Z}_k|}$$

such that

$$\left[\sum_{k=1}^{K} \log p(z_k = z_{i,k} \mid \mathbf{x})\right]_{i=1}^{|\mathcal{Z}|} = \mathbf{S}\boldsymbol{\ell}_{\text{ind}}(\mathbf{x}). \tag{13}$$

By the definition of the log-factorization residual,

$$[\log p(\mathbf{z} = \mathbf{z}_i \mid \mathbf{x})]_{i=1}^{|\mathcal{Z}|} = \mathbf{S}\boldsymbol{\ell}_{\text{ind}}(\mathbf{x}) + \mathbf{r}(\mathbf{x}). \tag{14}$$

Using Eq. (3),

$$\mathbf{f}_L(\mathbf{x}) = \mathbf{A}\,[\log p(\mathbf{z} = \mathbf{z}_i \mid \mathbf{x})]_{i=1}^{|\mathcal{Z}|} + \mathbf{b} + \mathbf{e}_L(\mathbf{x}). \tag{15}$$

Substituting Eq. (14) gives

$$\mathbf{f}_L(\mathbf{x}) = \mathbf{AS}\boldsymbol{\ell}_{\text{ind}}(\mathbf{x}) + \mathbf{b} + \mathbf{Ar}(\mathbf{x}) + \mathbf{e}_L(\mathbf{x}). \tag{16}$$

Defining

$$\widetilde{\mathbf{A}} := \mathbf{AS}$$

and

$$\widetilde{\mathbf{e}}_L(\mathbf{x}) := \mathbf{Ar}(\mathbf{x}) + \mathbf{e}_L(\mathbf{x}),$$

we obtain

$$\mathbf{f}_L(\mathbf{x}) = \widetilde{\mathbf{A}}\,[[\log p(z_1 \mid \mathbf{x})]_{z_1}; \ldots; [\log p(z_K \mid \mathbf{x})]_{z_K}] + \mathbf{b} + \widetilde{\mathbf{e}}_L(\mathbf{x}), \tag{17}$$

where

$$\widetilde{\mathbf{A}} \in \mathbb{R}^{m \times \sum_{k=1}^{K} |\mathcal{Z}_k|}.$$

We next bound the additional residual term $\mathbf{A}\mathbf{r}(\mathbf{x})$ by exploiting the structure of the mixing matrix $\mathbf{A}$ inherited from Liu et al. (2026). Recall from Liu et al. (2026) that

$$\mathbf{A} = (\widehat{\mathbf{L}}^\top)^{-1}\mathbf{L},$$

where $\widehat{\mathbf{L}}$ is invertible and $L_{j,i} = p(\mathbf{z} = \mathbf{z}_i \mid y_j) - p(\mathbf{z} = \mathbf{z}_i \mid y_0)$. Then, for each $j = 1, \ldots, m$,

$$|(\mathbf{L}\mathbf{r}(\mathbf{x}))_j| = \left| \sum_{i=1}^{|\mathcal{Z}|} \left[p(\mathbf{z} = \mathbf{z}_i \mid y_j) - p(\mathbf{z} = \mathbf{z}_i \mid y_0) \right] r_i(\mathbf{x}) \right| \tag{18}$$

$$\leq \sum_{i=1}^{|\mathcal{Z}|} |p(\mathbf{z} = \mathbf{z}_i \mid y_j) - p(\mathbf{z} = \mathbf{z}_i \mid y_0)|\,|r_i(\mathbf{x})| \tag{19}$$

$$\leq \max_i |r_i(\mathbf{x})| \sum_{i=1}^{|\mathcal{Z}|} |p(\mathbf{z} = \mathbf{z}_i \mid y_j) - p(\mathbf{z} = \mathbf{z}_i \mid y_0)| \tag{20}$$

$$= 2\ \mathrm{TV}\left(p(\mathbf{z} \mid y_j), p(\mathbf{z} \mid y_0)\right) \max_i |r_i(\mathbf{x})|. \tag{21}$$

Defining

$$\gamma := \max_{j=1,\ldots,m} \mathrm{TV}\left(p(\mathbf{z} \mid y_j), p(\mathbf{z} \mid y_0)\right), \tag{22}$$

we obtain

$$\|\mathbf{L}\mathbf{r}(\mathbf{x})\|_2 \leq 2\sqrt{m}\,\gamma \max_i |r_i(\mathbf{x})|. \tag{23}$$

Since $\mathbf{A} = (\widehat{\mathbf{L}}^\top)^{-1}\mathbf{L}$, we further have

$$\|\mathbf{A}\mathbf{r}(\mathbf{x})\|_2 \leq \|(\widehat{\mathbf{L}}^\top)^{-1}\|_2 \|\mathbf{L}\mathbf{r}(\mathbf{x})\|_2 \leq 2\sqrt{m}\,\|(\widehat{\mathbf{L}}^\top)^{-1}\|_2 \gamma \max_i |r_i(\mathbf{x})|. \tag{24}$$

This proves the result. □

D Proof of Corollary 3.2

Proof. For brevity, denote $[[\log p(z_1 \mid \mathbf{x})]_{z_1}; \ldots; [\log p(z_K \mid \mathbf{x})]_{z_K}]$ as $\boldsymbol{\ell}_{\mathrm{ind}}(\mathbf{x})$. By Eq. (6), the final output normalization satisfies

$$\mathbf{f}_L(\mathbf{x}) = \mathbf{G}\frac{\mathbf{f}_{L-1}(\mathbf{x}) - \mathbf{m}(\mathbf{f}_{L-1}(\mathbf{x}))}{s(\mathbf{f}_{L-1}(\mathbf{x}))} + \boldsymbol{\beta}. \tag{25}$$

Since $\mathbf{G}$ is invertible by assumption and $s(\mathbf{f}_{L-1}(\mathbf{x})) > 0$, we can rewrite this as

$$\mathbf{f}_{L-1}(\mathbf{x}) = \mathbf{m}(\mathbf{f}_{L-1}(\mathbf{x})) + s(\mathbf{f}_{L-1}(\mathbf{x}))\mathbf{G}^{-1}\left(\mathbf{f}_L(\mathbf{x}) - \boldsymbol{\beta}\right). \tag{26}$$

By Corollary 3.1, the final-layer representation satisfies

$$\mathbf{f}_L(\mathbf{x}) = \widetilde{\mathbf{A}}\boldsymbol{\ell}_{\mathrm{ind}}(\mathbf{x}) + \mathbf{b} + \widetilde{\mathbf{e}}_L(\mathbf{x}). \tag{27}$$

Substituting this into Eq. (26), we obtain

$$\mathbf{f}_{L-1}(\mathbf{x}) = \mathbf{m}(\mathbf{f}_{L-1}(\mathbf{x})) + s(\mathbf{f}_{L-1}(\mathbf{x}))\mathbf{G}^{-1}\left(\widetilde{\mathbf{A}}\boldsymbol{\ell}_{\mathrm{ind}}(\mathbf{x}) + \mathbf{b} - \boldsymbol{\beta} + \widetilde{\mathbf{e}}_L(\mathbf{x})\right). \tag{28}$$

Add and subtract the fixed reference statistics $\bar{\mathbf{m}}$ and $\bar{s}$. By the definitions

$$\bar{\mathbf{A}} := \bar{s}\,\mathbf{G}^{-1}\widetilde{\mathbf{A}} \tag{29}$$

and

$$\boldsymbol{\nu}(\mathbf{x}) := \mathbf{m}(\mathbf{f}_{L-1}(\mathbf{x})) - \bar{\mathbf{m}} + (s(\mathbf{f}_{L-1}(\mathbf{x})) - \bar{s})\,\mathbf{G}^{-1}\,(\mathbf{f}_L(\mathbf{x}) - \boldsymbol{\beta}) - \bar{\boldsymbol{\rho}}, \tag{30}$$

we can rewrite the previous display as

$$\mathbf{f}_{L-1}(\mathbf{x}) = \bar{\mathbf{A}}\boldsymbol{\ell}_{\mathrm{ind}}(\mathbf{x}) + \bar{\mathbf{m}} + \bar{s}\,\mathbf{G}^{-1}(\mathbf{b} - \boldsymbol{\beta}) + \bar{\boldsymbol{\rho}} + \bar{s}\,\mathbf{G}^{-1}\widetilde{\mathbf{e}}_L(\mathbf{x}) + \boldsymbol{\nu}(\mathbf{x}). \tag{31}$$

By the definition of ρ,

$$\|\boldsymbol{\nu}(\mathbf{x})\|_2 \le \rho. \tag{32}$$

Moreover,

$$\left\|\bar{s}\,\mathbf{G}^{-1}\widetilde{\mathbf{e}}_L(\mathbf{x})\right\|_2 \le |\bar{s}|\|\mathbf{G}^{-1}\|_2\|\widetilde{\mathbf{e}}_L(\mathbf{x})\|_2. \tag{33}$$

We now pull this pre-normalization representation back to an intermediate layer. By the residual update in Eq. (5), for $q = \ell, \dots, L-2$,

$$\mathbf{f}_{q+1}(\mathbf{x}) - \mathbf{f}_q(\mathbf{x}) = \Phi_q(\mathbf{f}_q(\mathbf{x})). \tag{34}$$

Summing from $q = \ell$ to $L-2$ gives

$$\mathbf{f}_{L-1}(\mathbf{x}) - \mathbf{f}_\ell(\mathbf{x}) = \sum_{q=\ell}^{L-2} \Phi_q(\mathbf{f}_q(\mathbf{x})), \tag{35}$$

and therefore

$$\mathbf{f}_\ell(\mathbf{x}) = \mathbf{f}_{L-1}(\mathbf{x}) - \sum_{q=\ell}^{L-2} \Phi_q(\mathbf{f}_q(\mathbf{x})). \tag{36}$$

Substituting Eq. (31) into Eq. (36) yields

$$\mathbf{f}_\ell(\mathbf{x}) = \bar{\mathbf{A}}\boldsymbol{\ell}_{\mathrm{ind}}(\mathbf{x}) + \bar{\mathbf{m}} + \bar{s}\,\mathbf{G}^{-1}(\mathbf{b} - \boldsymbol{\beta}) + \bar{\boldsymbol{\rho}} \tag{37}$$

$$- \sum_{q=\ell}^{L-2} \Phi_q(\bar{\mathbf{f}}_q) + \bar{s}\,\mathbf{G}^{-1}\widetilde{\mathbf{e}}_L(\mathbf{x}) + \boldsymbol{\nu}(\mathbf{x}) \tag{38}$$

$$- \sum_{q=\ell}^{L-2} \left[\Phi_q(\mathbf{f}_q(\mathbf{x})) - \Phi_q(\bar{\mathbf{f}}_q)\right]. \tag{39}$$

Using the definition

$$\mathbf{b}_\ell := \bar{\mathbf{m}} + \bar{s}\,\mathbf{G}^{-1}(\mathbf{b} - \boldsymbol{\beta}) + \bar{\boldsymbol{\rho}} - \sum_{q=\ell}^{L-2} \Phi_q(\bar{\mathbf{f}}_q), \tag{40}$$

we obtain

$$\mathbf{f}_\ell(\mathbf{x}) - \bar{\mathbf{A}}\boldsymbol{\ell}_{\mathrm{ind}}(\mathbf{x}) - \mathbf{b}_\ell \tag{41}$$

$$= \bar{s}\,\mathbf{G}^{-1}\widetilde{\mathbf{e}}_L(\mathbf{x}) + \boldsymbol{\nu}(\mathbf{x}) - \sum_{q=\ell}^{L-2} \left[\Phi_q(\mathbf{f}_q(\mathbf{x})) - \Phi_q(\bar{\mathbf{f}}_q)\right]. \tag{42}$$

Taking norms and applying the triangle inequality gives

$$\left\|\mathbf{f}_\ell(\mathbf{x}) - \bar{\mathbf{A}}\boldsymbol{\ell}_{\mathrm{ind}}(\mathbf{x}) - \mathbf{b}_\ell\right\|_2 \tag{43}$$

$$\le \left\|\bar{s}\,\mathbf{G}^{-1}\widetilde{\mathbf{e}}_L(\mathbf{x})\right\|_2 + \|\boldsymbol{\nu}(\mathbf{x})\|_2 + \sum_{q=\ell}^{L-2} \left\|\Phi_q(\mathbf{f}_q(\mathbf{x})) - \Phi_q(\bar{\mathbf{f}}_q)\right\|_2. \tag{44}$$

The first two terms are bounded by Eq. (33) and Eq. (32). For the residual-update terms, the reference-anchored Lipschitz condition gives

$$\left\|\Phi_q(\mathbf{f}_q(\mathbf{x})) - \Phi_q(\bar{\mathbf{f}}_q)\right\|_2 \le L_q \left\|\mathbf{f}_q(\mathbf{x}) - \bar{\mathbf{f}}_q\right\|_2 \le L_q R_q. \tag{45}$$

Therefore,

$$\left\|\mathbf{f}_\ell(\mathbf{x}) - \bar{\mathbf{A}}\boldsymbol{\ell}_{\mathrm{ind}}(\mathbf{x}) - \mathbf{b}_\ell\right\|_2 \le |\bar{s}|\|\mathbf{G}^{-1}\|_2\|\widetilde{\mathbf{e}}_L(\mathbf{x})\|_2 + \rho + \sum_{q=\ell}^{L-2} L_q R_q. \tag{46}$$

This proves Eq. (7). When $\ell = L-1$, all sums are empty, and the same argument gives the bound directly for the final pre-output-normalization representation. □

E JOINT POSTERIOR ESTIMATION VIA LINEAR PROBE

Corollary E.1. *Suppose that Eq. (3) holds, i.e.,*

$$\mathbf{f}(\mathbf{x}) \approx \mathbf{A}\boldsymbol{\ell}_{\text{joint}}(\mathbf{x}) + \mathbf{b}, \tag{47}$$

where

$$\boldsymbol{\ell}_{\text{joint}}(\mathbf{x}) = [\log p(\mathbf{z} = \mathbf{z}_i \mid \mathbf{x})]_{i=1}^{|\mathcal{Z}|}. \tag{48}$$

Assume that $\mathbf{A}$ *is left-invertible. Then there exists a multinomial linear probe whose softmax output recovers the joint posterior:*

$$\text{softmax}\left(\mathbf{W}\mathbf{f}(\mathbf{x}) + \mathbf{c}\right) \approx p(\mathbf{z} \mid \mathbf{x}). \tag{49}$$

Proof. Since $\mathbf{A}$ is left-invertible, there exists a linear map $\mathbf{W}$ such that

$$\mathbf{W}\mathbf{A} = \mathbf{I}. \tag{50}$$

Applying this readout to Eq. (3) gives

$$\mathbf{W}\mathbf{f}(\mathbf{x}) + \mathbf{c} \approx \mathbf{W}\mathbf{A}\boldsymbol{\ell}_{\text{joint}}(\mathbf{x}) + \mathbf{W}\mathbf{b} + \mathbf{c} \tag{51}$$

$$= \boldsymbol{\ell}_{\text{joint}}(\mathbf{x}) + \text{const}, \tag{52}$$

where const denotes a shared additive constant across all joint configurations.

The i-th recovered logit therefore satisfies

$$s_i(\mathbf{x}) \approx \log p(\mathbf{z} = \mathbf{z}_i \mid \mathbf{x}) + \text{const}. \tag{53}$$

Applying softmax removes this shared additive constant, yielding

$$\text{softmax}\left(\mathbf{W}\mathbf{f}(\mathbf{x}) + \mathbf{c}\right)_i \approx \frac{p(\mathbf{z} = \mathbf{z}_i \mid \mathbf{x})}{\sum_{j=1}^{|\mathcal{Z}|} p(\mathbf{z} = \mathbf{z}_j \mid \mathbf{x})} \tag{54}$$

$$= p(\mathbf{z} = \mathbf{z}_i \mid \mathbf{x}). \tag{55}$$

Hence,

$$\text{softmax}\left(\mathbf{W}\mathbf{f}(\mathbf{x}) + \mathbf{c}\right) \approx p(\mathbf{z} \mid \mathbf{x}). \tag{56}$$

□

F BINARY LOG-POSTERIOR ESTIMATION VIA LINEAR PROBING

Corollary F.1. *Suppose that Corollary 3.1 holds, and consider a binary concept* $z_k \in \{0, 1\}$ *with counterfactual examples differing only in* z_k*. Assume that its marginal log-posterior contrast is linearly readable from the LLM representation, i.e., there exists a linear score* $s_k(\mathbf{x}) = \mathbf{w}_k^\top \mathbf{f}_L(\mathbf{x}) + a_k$*, such that* $s_k(\mathbf{x}) \approx \log p(z_k = 1 \mid \mathbf{x}) - \log p(z_k = 0 \mid \mathbf{x})$*. Then, at the population optimum of binary cross-entropy classification,* $\sigma(s_k(\mathbf{x})) \approx p(z_k = 1 \mid \mathbf{x})$*.*

Proof. For a binary classifier with score $s_k(\mathbf{x})$, the population cross-entropy loss conditioned on $\mathbf{x}$ is

$$\mathcal{L}_{\mathbf{x}} = -p_k(\mathbf{x}) \log \sigma(s_k(\mathbf{x})) - \big(1 - p_k(\mathbf{x})\big) \log\big(1 - \sigma(s_k(\mathbf{x}))\big),$$

where

$$p_k(\mathbf{x}) := p(z_k = 1 \mid \mathbf{x}).$$

Binary cross-entropy is minimized when the predicted probability equals the true conditional probability, giving

$$\sigma(s_k^*(\mathbf{x})) = p(z_k = 1 \mid \mathbf{x}).$$

Equivalently,

$$s_k^*(\mathbf{x}) = \log \frac{p(z_k = 1 \mid \mathbf{x})}{p(z_k = 0 \mid \mathbf{x})}.$$

Under the linear-readability condition above, this optimal log-odds is represented by a linear probe on $\mathbf{f}_L(\mathbf{x})$, up to the approximation residual inherited from Corollary 3.1, which proves the result. □

G Experimental Details for Verifying Corollary 3.1

Corollary 3.1 shows that the joint log-posterior can be decomposed into a component constructed from marginal log-posteriors and a log-factorization residual $\mathbf{r}(\mathbf{x})$, which induces the representation-level interaction residual $\mathbf{A}\mathbf{r}(\mathbf{x})$. Since the true posterior $p(\mathbf{z} \mid \mathbf{x})$ is unavailable for real-world text, we estimate the joint posterior from frozen LLM representations and empirically examine the magnitude of this residual. Motivated by Eq. (3) and the result of Liu et al. (2026), we estimate the joint concept posterior using a linear probe on the LLM representations, see Sec. E for details.

We conduct experiments on the E2E dataset (Novikova et al., 2017), which contains natural-language restaurant descriptions. We use four discrete attributes as the annotated concepts: `eatType`, `area`, `familyFriendly`, and a discretized `customerRating`. Samples with missing or ambiguous values for any of these attributes are removed, leaving 24 observed joint concept configurations. For each input $\mathbf{x}$, we freeze the LLM and extract the final-layer representations $\mathbf{f}_L(\mathbf{x})$. We split the data into 70% training, 10% validation, and 20% test examples, and repeat the experiment over three random splits. For each split, we train a multinomial linear probe to predict the joint concept configuration

$$\mathbf{z} = (z_{\text{eatType}}, z_{\text{area}}, z_{\text{familyFriendly}}, z_{\text{rating}}). \tag{57}$$

Motivated by Corollary E.1, we estimate the joint posterior using a multiclass linear probe applied to the frozen final-layer LLM representations. The probe is trained on the training set, with its regularization hyperparameters selected on the validation set. Specifically, we implement a regularized multiclass linear classifier using the scikit-learn Python package (Pedregosa et al., 2011). We apply temperature scaling, with the temperature selected on the validation set by minimizing negative log-likelihood. The test split is used only for final evaluation. Following Corollary E.1, we use the softmax output of the probe as an empirical estimate of the joint posterior,

$$\widehat{p}_J(\mathbf{z} \mid \mathbf{x}). \tag{58}$$

Based on this, we obtain the marginal posterior of each individual concept by marginalizing over the remaining concepts:

$$\widehat{p}_k(z_k = a \mid \mathbf{x}) = \sum_{\mathbf{z}:\, z_k = a} \widehat{p}_J(\mathbf{z} \mid \mathbf{x}). \tag{59}$$

We then construct the product of the concept-wise marginal posteriors as

$$\widehat{p}_M(\mathbf{z}_i \mid \mathbf{x}) = \prod_{k=1}^{K} \widehat{p}_k(z_k = z_{i,k} \mid \mathbf{x}), \tag{60}$$

and compute the empirical log-factorization residual

$$\widehat{r}_i(\mathbf{x}) = \log \widehat{p}_J(\mathbf{z}_i \mid \mathbf{x}) - \log \widehat{p}_M(\mathbf{z}_i \mid \mathbf{x}). \tag{61}$$

Collecting these terms into

$$\widehat{\mathbf{r}}(\mathbf{x}) = [\widehat{r}_i(\mathbf{x})]_i, \tag{62}$$

we quantify the resulting substitution error on test split by the relative residual

$$\widehat{\delta}_r := \sqrt{\frac{\sum_{\mathbf{x}} \|\widehat{\mathbf{r}}(\mathbf{x})\|_2^2}{\sum_{\mathbf{x}} \|[\log \widehat{p}_J(\mathbf{z}_i \mid \mathbf{x})]_i\|_2^2}}, \tag{63}$$

which measures the magnitude of the log-factorization residual relative to the joint log-posterior signal.

To estimate the mixing matrix $\mathbf{A}$, we follow its structural form in Corollary 3.1. Using only the training split, we select a set of sufficiently supported anchor tokens $y_0, \ldots, y_m$ whose unembedding-vector differences satisfy the diversity condition. For each anchor token, we estimate its conditional concept posterior using the annotated E2E configurations and the model's next-token probabilities:

$$\widehat{p}(\mathbf{z}_i \mid y) = \frac{\sum_{\mathbf{x} \in \mathcal{D}_{\text{train}}} \mathbb{1}\{\mathbf{z}(\mathbf{x}) = \mathbf{z}_i\} p_\theta(y \mid \mathbf{x})}{\sum_{\mathbf{x} \in \mathcal{D}_{\text{train}}} p_\theta(y \mid \mathbf{x})}. \tag{64}$$

We then construct the unembedding and posterior contrast matrices

$$\widehat{\mathbf{L}}_{\text{emb}} = [\mathbf{g}(y_j) - \mathbf{g}(y_0)]_{j=1}^{m}, \qquad [\widehat{\mathbf{L}}_{\text{post}}]_{j,i} = \widehat{p}(\mathbf{z}_i \mid y_j) - \widehat{p}(\mathbf{z}_i \mid y_0), \tag{65}$$

and estimate the mixing matrix as

$$\widehat{\mathbf{A}} = (\widehat{\mathbf{L}}_{\text{emb}}^{\top})^{-1} \widehat{\mathbf{L}}_{\text{post}}. \tag{66}$$

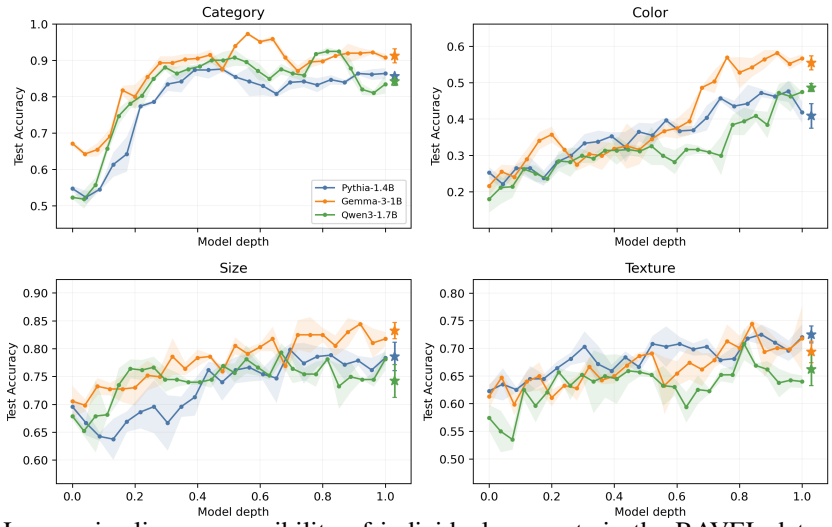

Figure 6: Layer-wise linear accessibility of individual concepts in the RAVEL dataset. Across attributes and models, concept generally becomes more linearly accessible toward later layers.

H Experimental Details for Verifying Corollary 3.2

We use the RAVEL `physical_object` domain as a factual multi-concept benchmark. Each example corresponds to a physical-object entity and is formatted as a short factual prompt containing only the entity name, e.g., "The physical object is `American football`." Each entity is associated with four annotated concepts: Category, Color, Size, and Texture. We follow the official RAVEL entity-level split, with disjoint entities across the training, validation, and test sets. The resulting split contains 276 training entities, 150 validation entities, and 137 test entities. The training split is used to estimate all reference statistics and to fit the linear probes, the validation split is used for probe hyperparameter selection, and the test split is used exclusively for reporting the final results. For each LLM, we perform a forward pass over all examples and cache the last-token residual-stream representation after every transformer block. We also retain the final pre-output-normalization residual state $\mathbf{f}_{L-1}$ and the output-normalized readout representation $\mathbf{f}_L$.

Normalization-Statistics Variation. To evaluate the empirical counterpart of the normalization term ρ, we compute the example-specific output-normalization statistics from the final pre-normalization residual state $\mathbf{f}_{L-1}$. Fixed reference centering and scaling statistics are estimated exclusively from the training split. For each test example, we measure the input-dependent deviation from the normalization obtained using these fixed reference statistics. We further center this deviation by subtracting its mean over the training split, thereby removing the input-independent component that can be absorbed into the representation bias. We report the mean norm of the resulting variation on the test split, normalized by the mean norm of the centered final pre-normalization representation. It measures the input-dependent variation introduced by the final output normalization.

Residual-tail contribution. For the residual-tail term, we avoid directly estimating the potentially loose Lipschitz constants in $\sum_{q=\ell}^{L-2} L_q R_q$. Instead, at each layer ℓ, we measure its empirical counterpart using the cumulative residual tail between the layer representation and the final pre-normalization residual state, $\mathbf{T}_\ell(\mathbf{x}) = \mathbf{f}_{L-1}(\mathbf{x}) - \mathbf{f}_\ell(\mathbf{x})$. We center this quantity using its mean over the training split, thereby removing the input-independent component that can be absorbed into the layer-specific bias. On the test split, we report the average norm of the centered residual tail, normalized by the average norm of the centered representation at layer ℓ.

Layer-wise concept accessibility. Finally, we train linear probes at every layer to predict each of the four concepts. For each layer and concept, the probe is fitted using the training split, its regularization strength is selected using the validation split, and its accuracy is evaluated on the test data. The shaded region in the figure denotes one standard deviation across the three probe seeds. As shown in Fig. 6, the four concepts generally become more linearly accessible toward later layers across the three LLMs, consistent with Corollary 3.2.

I Training Details for Concept Extraction Experiments

Activation Data For all experiments, we use publicly available pretrained LLM checkpoints downloaded from Hugging Face,https://huggingface.co/, including Pythia-70M, Pythia-1.4B, and Pythia-2.8B (Biderman et al., 2023), Gemma-3-1B (Team et al., 2025), and Qwen3-1.7B (Team, 2025). All LLM parameters remain frozen. For each model, we extract representations from approximately the first 200 million non-padding tokens in the training split of the deduplicated Pile dataset (Gao et al., 2020),https://huggingface.co/datasets/EleutherAI/the_pile_deduplicated. Specifically, for every token, we record the final output-normalized hidden representation. This choice is motivated by Corollaries 3.1 and 3.2: the posterior-coordinate approximation is defined at the final readout, while the layer-wise bound shows that this approximation should generally become more accurate toward later layers as the residual-tail contribution decreases. The resulting cached representations are used as the training data for the proposed sparse ConCa and all SAE baselines. The dimensionality of the training representations depends on the LLM backbone. In every case, the learned dictionary contains 32,768 features.

Optimization. We stream the cached activation shards during training, randomly shuffling both the shard order and the representations within each shard. All methods use a batch size of 10,000 activation vectors and a learning rate of 10^{-4}, with a linear learning-rate warmup over the first 200 optimization steps. For methods with an explicit sparsity penalty, the penalty is warmed up over the first 400 steps. Decoder directions are constrained to have unit norm to remove the scaling ambiguity between feature activations and decoder vectors. Models are trained through step 20,000. We use the same hyperparameters for a given method across all five LLM backbones.

SAE baselines. Top-k SAE retains the $k = 32$ largest nonnegative feature activations independently for each token. Batch-Top-k applies the sparsity budget jointly across each training batch, using a budget corresponding to an average of $k = 32$ active features per token. PAnneal uses a ReLU SAE with an L_p^p sparsity objective. Its initial sparsity coefficient is 0.1. The exponent is held at $p = 1$ during the early stage of training and is annealed from 1 to 0.2 after step 15,000 using ten equally spaced updates. Dead features are resampled every 5,000 steps. Structured SAE uses the structured encoder proposed in Liu et al. (2026), consisting of a main ReLU encoder and an additional low-rank-regularized encoder branch. The outputs of the two branches are summed before reconstruction. We use an initial sparsity coefficient of 0.1, anneal the L_p exponent from 1 toward 0 after step 15,000, and weight the low-rank regularizer by 0.1 relative to the sparsity coefficient, as suggested by Liu et al. (2026). The remaining optimization settings are shared with PAnneal.

ConCa. For the proposed sparse ConCa configuration, the encoder coordinates are normalized using LayerNorm before reconstruction. Following the posterior-coordinate interpretation, the sparsity regularizer is applied to the transformed scores $a_j(\mathbf{x}) = \exp(z_j(\mathbf{x}))$. We use an L_1 penalty on these transformed scores with coefficient 1×10^{-4}; equivalently, the L_p exponent is fixed at $p = 1$ throughout training. The clipping threshold, sparsity coefficient, normalization architecture, and all optimization hyperparameters are fixed across the five backbones. See model choice experiments in Sec. L for more details.

We implement the SAE baselines using the open-source `dictionary_learning` library, https://github.com/saprmarks/dictionary_learning, which provides the training implementations for Top-k, Batch-Top-k, and PAnneal SAEs. We extend the same training framework to implement Structured SAE and ConCa, ensuring that all methods use the same activation data, dictionary width, batching pipeline, and optimization budget. All experiments are conducted on NVIDIA A100 GPUs with 40 GB of memory.

Table 4: Standard SAEBench reconstruction results across LLM backbones. We report mean $\pm$ standard deviation over three runs. Higher cosine similarity and lower MSE indicate better reconstruction fidelity.

Model	Method	Cosine similarity ↑	MSE ↓
Pythia-70M	TopK	0.7837 ± 0.0045	54.9802 ± 1.0279
	BatchTopK	0.7528 ± 0.0103	106.6620 ± 6.2019
	Panneal	0.9812 ± 0.0017	5.4443 ± 0.4324
	StructuredSAE	0.9825 ± 0.0012	28.6704 ± 5.0495
	ConCa	$\mathbf{0.9998 \pm 0.000004}$	$\mathbf{2.7660 \pm 0.0350}$
Pythia-1.4B	TopK	0.8949 ± 0.000075	2.4232 ± 0.0017
	BatchTopK	0.8931 ± 0.000062	2.4689 ± 0.0010
	Panneal	0.9967 ± 0.000076	0.0839 ± 0.0020
	StructuredSAE	0.9996 ± 0.00014	0.0108 ± 0.0034
	ConCa	$\mathbf{0.9999 \pm 0.000042}$	$\mathbf{0.0058 \pm 0.0011}$
Pythia-2.8B	TopK	0.8818 ± 0.000053	2.6448 ± 0.0010
	BatchTopK	0.8798 ± 0.00011	2.6921 ± 0.0024
	Panneal	0.9960 ± 0.000083	0.0974 ± 0.0021
	StructuredSAE	0.9992 ± 0.00081	0.0188 ± 0.0095
	ConCa	$\mathbf{0.9999 \pm 0.000021}$	$\mathbf{0.0048 \pm 0.0005}$

J Evaluation Details of Concept Extraction Experiments

SAEBench (Karvonen et al., 2025) provides a comprehensive framework for evaluating SAEs from two complementary perspectives. The first concerns *representation fidelity*: whether the learned feature space preserves enough information to reconstruct the original LLM representation. The second concerns *concept quality*: whether learned features correspond to disentangled concepts, respond selectively when the corresponding concept changes, and preserve concept information.

We retain the original SAEBench reconstruction evaluation, reporting cosine similarity and mean squared error (MSE) between the original LLM representation and its reconstruction. Following the standard SAEBench core protocol (Karvonen et al., 2025), we evaluate on token sequences sampled from the `OpenWebText` dataset (Gokaslan & Cohen, 2019). Table 4 shows the results.

To connect the concept-oriented part of SAEBench more directly to Corollaries 3.1 and 3.2, we adapt its three main evaluation perspectives, i.e., alignment, sensitivity, and information recovery, to the marginal posteriors. The resulting metrics are termed *Posterior Alignment*, *Posterior Sensitivity*, and *Posterior Recovery*. Here, Posterior Alignment and Posterior Sensitivity, rely on counterfactual pair data that differ in only a single concept of interest while keeping all other aspects unchanged. Before introducing the three metrics, we construct the following 4 datasets as follows.

RAVEL. We use the Physical Object subset of RAVEL dataset (Huang et al., 2024), in which each entity is annotated with four attributes: category, color, size, and texture. For each attribute, we enumerate all unordered pairs of its possible values. Two entities form an exact counterfactual pair if they differ in the selected attribute values while agreeing exactly on the other three attributes. To ensure sufficient statistical support, we retain only binary pairs containing at least 100 exact pairs. This produces 43 binary concept pairs: 3 category pairs, 37 color pairs, 2 size pairs, and 1 texture pair. The resulting dataset contains 22,715 paired samples.

Bias in Bios. We use the Bias in Bios dataset (De-Arteaga et al., 2019) and select the eight professions: professor, nurse, architect, journalist, surgeon, psychologist, attorney, and teacher. After removing exact duplicate biographies, we sample 250 examples from each profession-gender combination, resulting in 4,000 biographies. We construct all $\binom{8}{2} = 28$ binary profession concepts while keeping gender fixed, together with one binary gender concept while keeping profession fixed. This gives 29 binary concepts and 16,000 matched sample pairs in total.

Amazon Reviews. We use the Amazon Reviews dataset (Ni et al., 2019) and select eight product categories used by SAEBench: Books, CDs and Vinyl, Software, Electronics, Pet Supplies, Office Products, Industrial and Scientific, and Toys and Games. We retain only 1-star and 5-star reviews

Table 5: Concept posterior alignment across four counterfactual datasets. Results are mean ± standard deviation over three runs, higher is better.

Backbone	TopK	BatchTopK	PAnneal	StructuredSAE	ConCa
RAVEL					
Pythia-70M	0.3873 ± 0.0022	0.3165 ± 0.0040	0.5796 ± 0.0024	0.5956 ± 0.0002	**0.6385 ± 0.0022**
Pythia-1.4B	0.4058 ± 0.0070	0.3534 ± 0.0049	0.6277 ± 0.0070	0.6313 ± 0.0011	**0.6636 ± 0.0029**
Pythia-2.8B	0.3889 ± 0.0099	0.3184 ± 0.0020	0.6204 ± 0.0028	0.6380 ± 0.0009	**0.6581 ± 0.0042**
Qwen3-1.7B	0.4307 ± 0.0190	0.3795 ± 0.0087	0.6286 ± 0.0142	0.6648 ± 0.0023	**0.6818 ± 0.0051**
Gemma-3-1B	0.4287 ± 0.0073	0.3951 ± 0.0065	0.6522 ± 0.0211	0.6804 ± 0.0054	**0.6955 ± 0.0045**
Bias in Bios					
Pythia-70M	0.3389 ± 0.0027	0.3234 ± 0.0048	0.5174 ± 0.0117	0.5640 ± 0.0005	**0.6065 ± 0.0057**
Pythia-1.4B	0.4361 ± 0.0003	0.4052 ± 0.0066	0.5847 ± 0.0098	0.5977 ± 0.0038	**0.6671 ± 0.0042**
Pythia-2.8B	0.4168 ± 0.0055	0.4022 ± 0.0044	0.5720 ± 0.0104	0.6082 ± 0.0035	**0.6737 ± 0.0052**
Qwen3-1.7B	0.4139 ± 0.0128	0.3947 ± 0.0051	0.5652 ± 0.0125	0.5797 ± 0.0096	**0.6089 ± 0.0081**
Gemma-3-1B	0.4325 ± 0.0036	0.4200 ± 0.0062	0.5669 ± 0.0185	0.6406 ± 0.0055	**0.6777 ± 0.0017**
Amazon Reviews					
Pythia-70M	0.2685 ± 0.0041	0.2507 ± 0.0028	0.4144 ± 0.0065	0.4776 ± 0.0003	**0.5232 ± 0.0056**
Pythia-1.4B	0.3045 ± 0.0039	0.2991 ± 0.0011	0.4998 ± 0.0105	0.5025 ± 0.0068	**0.5933 ± 0.0030**
Pythia-2.8B	0.2985 ± 0.0026	0.2959 ± 0.0067	0.4858 ± 0.0093	0.4852 ± 0.0095	**0.5970 ± 0.0060**
Qwen3-1.7B	0.2849 ± 0.0044	0.2926 ± 0.0049	**0.4776 ± 0.0190**	0.4463 ± 0.0037	0.4510 ± 0.0028
Gemma-3-1B	0.3014 ± 0.0046	0.3070 ± 0.0062	0.4784 ± 0.0170	0.4751 ± 0.0030	**0.5514 ± 0.0027**
Park et al.					
Pythia-70M	0.5507 ± 0.0098	0.5206 ± 0.0085	0.6514 ± 0.0042	0.6806 ± 0.0001	**0.7254 ± 0.0051**
Pythia-1.4B	0.6773 ± 0.0061	0.6647 ± 0.0036	0.7367 ± 0.0051	0.7345 ± 0.0019	**0.7857 ± 0.0021**
Pythia-2.8B	0.6905 ± 0.0080	0.6884 ± 0.0052	0.7572 ± 0.0030	0.7506 ± 0.0016	**0.8008 ± 0.0026**
Qwen3-1.7B	0.5897 ± 0.0089	0.5792 ± 0.0109	0.6356 ± 0.0043	0.6828 ± 0.0042	**0.7362 ± 0.0008**
Gemma-3-1B	0.5486 ± 0.0006	0.5498 ± 0.0084	0.6317 ± 0.0058	0.7371 ± 0.0014	**0.8277 ± 0.0066**

and remove exact duplicate review texts. We then sample 250 reviews from each category–rating combination, resulting in 4,000 reviews. We construct all $\binom{8}{2} = 28$ binary category concepts while keeping the rating fixed, together with one binary rating concept while keeping the product category fixed. This gives 29 binary concepts and 16,000 matched sample pairs.

Park et al. Counterfactual Pairs. We additionally use the 27 high-precision counterfactual concepts introduced by Park et al. (2023). Each concept is represented by multiple word pairs that differ primarily in a single linguistic or semantic property. These concepts cover a diverse range of transformations, including verb inflections, adjective transformations, semantic relations, morphological changes, and cross-lingual translations. The number of paired samples varies across concepts, ranging from 4 to 228. These carefully controlled pairs provide an additional benchmark for evaluating whether learned features capture concept-specific posterior information.

Given the evaluation dataset above, we report the following metrics.

Posterior Alignment: Mean Correlation Coefficient Posterior Alignment evaluates whether an individual learned feature corresponds to a specific concept-wise log-posterior coordinate, rather than merely carrying information that is predictive of the concept. To quantify this correspondence, we use the Mean Correlation Coefficient (MCC). Algorithm 1 summarizes the procedure: for each of the d_c counterfactual concept pairs, we first estimate the corresponding log-posterior using a supervised linear classifier. The same inputs are then passed through the trained SAE or ConCa to extract latent features, which are stacked into a feature matrix. We compute the Pearson correlation between each SAE feature and the corresponding log-posterior, and solve the assignment problem using the Hungarian algorithm to account for permutation indeterminacy. The MCC across all concepts is reported as the final evaluation score. The detailed results can be found in Table 5.

Posterior Sensitivity: Counterfactual Sensitivity Posterior Sensitivity evaluates whether a feature aligned with a given concept-posterior coordinate responds specifically to changes in that concept. This complements Posterior Alignment by assessing the selectivity of the learned feature under

Algorithm 1 Evaluation of SAE/ConCa Concepts via Supervised Linear Classification

Require: Trained SAE/ConCa, d_c binary concepts with corresponding counterfactual datasets
Ensure: Mean Pearson correlation between learned features and supervised log-posteriors
Step 1: Obtain supervised log-posteriors
for $i = 1$ **to** d_c **do**
  Train LogisticRegression for the i-th binary concept
  Compute the log-posterior $\boldsymbol{\ell}_i = [\log \widehat{p}(z_i = 1 \mid \mathbf{x})]$ on all evaluation endpoints
end for
Step 2: Extract latent features
for $i = 1$ **to** d_c **do**
  Pass the same evaluation endpoints through the trained SAE/ConCa
  Collect latent feature activations $\widehat{\mathbf{Z}}_i \in \mathbb{R}^{N_i \times D}$
end for
Step 3: Compute correlation matrix
for $i = 1$ **to** d_c **do**
  for $d = 1$ **to** D **do**
    $R_{i,d} = \mathrm{PCC}(\boldsymbol{\ell}_i, \widehat{\mathbf{Z}}_{i,:,d})$
  end for
end for
Step 4: Solve assignment problem
Apply the Hungarian algorithm to $\mathbf{R}$ to obtain the optimal concept–feature assignment π
Extract the assigned correlations $\{R_{i,\pi(i)}\}_{i=1}^{d_c}$
Step 5: Aggregate metric
Report $\frac{1}{d_c}\sum_{i=1}^{d_c} R_{i,\pi(i)}$

controlled concept changes. To quantify this property, we introduce a counterfactual sensitivity gap that compares the response of the matched feature to the target concept against its responses to off-target concepts.

Specifically, for each binary concept i, we train a linear probe and define $\ell_i(\mathbf{x}) = \log \widehat{p}(z_i = 1 \mid \mathbf{x})$ as the estimated log-posterior. We denote the features extracted by the evaluated SAE or ConCa as $\hat{\mathbf{z}}(\mathbf{x}) = \mathrm{Enc}(\mathbf{f}(\mathbf{x}))$. Let $\pi(i)$ denote the feature matched to concept i in the Posterior Alignment evaluation. For each counterfactual pair $(\mathbf{x}_a, \mathbf{x}_b)$ associated with concept i, we compute

$$\Delta\ell_i = \ell_i(\mathbf{x}_b) - \ell_i(\mathbf{x}_a), \qquad \Delta\hat{z}_d = \hat{z}_d(\mathbf{x}_b) - \hat{z}_d(\mathbf{x}_a). \tag{67}$$

Here, $\Delta\ell_i$ measures the change in the estimated log-posterior across the counterfactual pair, while $\Delta\hat{z}_d$ measures the corresponding change in learned feature d.

The target sensitivity is defined as

$$T_i = \left|\mathrm{PCC}\left(\Delta\ell_i, \Delta\hat{z}_{\pi(i)}\right)\right|, \tag{68}$$

where the Pearson Correlation Coefficient (PCC) is computed across all paired samples associated with concept i. The off-target sensitivity O_i is the average absolute correlation between $\Delta\ell_i$ and the features matched to concepts other than i. We report

$$\text{Sensitivity} = \frac{1}{d_c}\sum_{i=1}^{d_c}\left(T_i - O_i\right), \tag{69}$$

where d_c denotes the number of evaluated binary concepts. A high score means that the feature assigned to concept i follows changes in its corresponding log-posterior while remaining insensitive to unrelated concept changes. See Alg. 2 for implementation, and Table 6 for the results.

Posterior Recovery. Posterior Recovery evaluates whether the learned representation preserves the concept-posterior information contained in the original LLM representation. For an input $\mathbf{x}$, let

$$\widehat{\mathbf{f}}(\mathbf{x}) = \mathrm{Dec}\left(\mathrm{Enc}(\mathbf{f}(\mathbf{x}))\right) \tag{70}$$

Table 6: Concept posterior sensitivity across four counterfactual datasets. We report mean ± standard deviation over three runs; higher is better.

Backbone	TopK	BatchTopK	PAnneal	StructuredSAE	ConCa
RAVEL					
Pythia-70M	0.1318 ± 0.0075	0.1213 ± 0.0075	0.2462 ± 0.0017	0.2717 ± 0.0023	**0.3384 ± 0.0040**
Pythia-1.4B	0.1116 ± 0.0125	0.0798 ± 0.0112	0.2050 ± 0.0206	0.2292 ± 0.0032	**0.2704 ± 0.0099**
Pythia-2.8B	0.0885 ± 0.0095	0.0559 ± 0.0151	0.1270 ± 0.0177	0.1939 ± 0.0056	**0.2472 ± 0.0059**
Qwen3-1.7B	0.0927 ± 0.0206	0.0489 ± 0.0127	0.1742 ± 0.0043	0.2430 ± 0.0077	**0.2757 ± 0.0260**
Gemma-3-1B	0.0799 ± 0.0105	0.0645 ± 0.0031	0.1593 ± 0.0084	**0.2513 ± 0.0078**	0.2477 ± 0.0162
Bias in Bios					
Pythia-70M	0.0145 ± 0.0037	0.0235 ± 0.0381	0.1898 ± 0.0398	0.2308 ± 0.0014	**0.3036 ± 0.0315**
Pythia-1.4B	0.0678 ± 0.0174	0.0479 ± 0.0058	0.1255 ± 0.0015	0.2203 ± 0.0214	**0.2926 ± 0.0371**
Pythia-2.8B	0.0406 ± 0.0031	0.0280 ± 0.0083	0.0714 ± 0.0124	0.1989 ± 0.0138	**0.3236 ± 0.0148**
Qwen3-1.7B	0.0358 ± 0.0035	0.0290 ± 0.0049	0.0873 ± 0.0059	0.1891 ± 0.0108	**0.2530 ± 0.0342**
Gemma-3-1B	-0.0098 ± 0.0106	0.0022 ± 0.0079	0.0846 ± 0.0300	0.2483 ± 0.0363	**0.3325 ± 0.0219**
Amazon Reviews					
Pythia-70M	0.0475 ± 0.0041	0.0226 ± 0.0114	0.1366 ± 0.0016	0.1939 ± 0.0030	**0.2219 ± 0.0287**
Pythia-1.4B	-0.0060 ± 0.0148	-0.0012 ± 0.0174	0.0072 ± 0.0151	0.1575 ± 0.0139	**0.2526 ± 0.0089**
Pythia-2.8B	0.0016 ± 0.0334	-0.0059 ± 0.0104	0.0024 ± 0.0148	0.1342 ± 0.0069	**0.2595 ± 0.0168**
Qwen3-1.7B	0.0162 ± 0.0125	0.0150 ± 0.0135	0.0669 ± 0.0115	0.1197 ± 0.0023	**0.1749 ± 0.0438**
Gemma-3-1B	0.0064 ± 0.0128	-0.0169 ± 0.0169	0.0917 ± 0.0085	0.1416 ± 0.0186	**0.2653 ± 0.0138**
Park et al.					
Pythia-70M	0.0379 ± 0.0022	0.0359 ± 0.0034	0.1227 ± 0.0163	0.1611 ± 0.0006	**0.2265 ± 0.0355**
Pythia-1.4B	0.1811 ± 0.0266	0.1812 ± 0.0162	0.1131 ± 0.0082	0.2155 ± 0.0047	**0.2946 ± 0.0055**
Pythia-2.8B	0.1410 ± 0.0307	0.1635 ± 0.0208	0.1293 ± 0.0189	0.1972 ± 0.0136	**0.3283 ± 0.0172**
Qwen3-1.7B	0.1566 ± 0.0143	0.1433 ± 0.0264	0.1318 ± 0.0169	0.1867 ± 0.0079	**0.2628 ± 0.0048**
Gemma-3-1B	0.2373 ± 0.0025	0.2341 ± 0.0188	0.3141 ± 0.0035	0.3923 ± 0.0064	**0.4948 ± 0.0209**

Algorithm 2 Evaluation of Posterior Sensitivity

Require: Trained SAE/ConCa, d_c binary concepts, paired samples, frozen linear probes, and feature assignment π from Posterior Alignment

Ensure: Posterior Sensitivity score

for $i = 1$ **to** d_c **do**
  for each paired sample $(\mathbf{x}_a, \mathbf{x}_b)$ associated with concept i **do**
    Compute log-posterior change $\Delta\ell_i = \ell_i(\mathbf{x}_b) - \ell_i(\mathbf{x}_a)$
    Extract features $\hat{\mathbf{z}}(\mathbf{x}_a) = \mathrm{Enc}(\mathbf{f}(\mathbf{x}_a))$, $\hat{\mathbf{z}}(\mathbf{x}_b) = \mathrm{Enc}(\mathbf{f}(\mathbf{x}_b))$
    Compute feature changes $\Delta\hat{\mathbf{z}} = \hat{\mathbf{z}}(\mathbf{x}_b) - \hat{\mathbf{z}}(\mathbf{x}_a)$
  end for
  Compute target sensitivity

$$T_i = \left|\mathrm{PCC}\left(\Delta\ell_i, \Delta\hat{z}_{\pi(i)}\right)\right|$$

  Compute off-target sensitivity

$$O_i = \frac{1}{d_c - 1} \sum_{j \neq i} \left|\mathrm{PCC}\left(\Delta\ell_i, \Delta\hat{z}_{\pi(j)}\right)\right|$$

  Compute sensitivity gap $G_i = T_i - O_i$
end for
return $\mathrm{Sensitivity} = \dfrac{1}{d_c} \sum_{i=1}^{d_c} G_i$

denote the reconstruction produced by the evaluated SAE or ConCa. For each concept i, we apply the same frozen linear probe used above to both the original and reconstructed representations, yielding the estimated log-posteriors $\ell_i(\mathbf{x})$ and $\widehat{\ell}_i(\mathbf{x})$, respectively.

Table 7: Concept posterior recovery across four counterfactual datasets. We report the squared Pearson correlation (PCC^2) between positive-class log-posteriors estimated from the original and reconstructed LLM representations. Results are mean $\pm$ standard deviation over three runs; higher is better.

Backbone	TopK	BatchTopK	PAnneal	StructuredSAE	ConCa
RAVEL					
Pythia-70M	0.5891 ± 0.0109	0.3936 ± 0.0054	0.9899 ± 0.0025	**0.9979 ± 0.0019**	0.9967 ± 0.0017
Pythia-1.4B	0.5720 ± 0.0087	0.4452 ± 0.0128	0.9972 ± 0.0002	**0.9999 ± 0.0000**	0.9999 ± 0.0000
Pythia-2.8B	0.5074 ± 0.0164	0.3516 ± 0.0104	0.9972 ± 0.0006	0.9994 ± 0.0001	**0.9998 ± 0.0000**
Qwen3-1.7B	0.5585 ± 0.0274	0.4794 ± 0.0231	0.9966 ± 0.0017	**0.9999 ± 0.0000**	0.9995 ± 0.0002
Gemma-3-1B	0.2796 ± 0.0205	0.2353 ± 0.0226	**0.9949 ± 0.0018**	0.8655 ± 0.0871	0.9280 ± 0.0564
Bias in Bios					
Pythia-70M	0.8740 ± 0.0043	0.8118 ± 0.0012	0.9938 ± 0.0018	**0.9997 ± 0.0003**	0.9994 ± 0.0003
Pythia-1.4B	0.9113 ± 0.0020	0.8834 ± 0.0052	0.9989 ± 0.0001	**0.9999 ± 0.0000**	0.9995 ± 0.0000
Pythia-2.8B	0.9008 ± 0.0009	0.8851 ± 0.0033	0.9988 ± 0.0001	**0.9999 ± 0.0001**	0.9998 ± 0.0000
Qwen3-1.7B	0.8847 ± 0.0111	0.8573 ± 0.0023	0.9987 ± 0.0007	**0.9999 ± 0.0000**	0.9997 ± 0.0000
Gemma-3-1B	0.8390 ± 0.0050	0.8025 ± 0.0129	**0.9989 ± 0.0008**	0.9977 ± 0.0009	0.9986 ± 0.0009
Amazon Reviews					
Pythia-70M	0.8969 ± 0.0018	0.8603 ± 0.0028	0.9945 ± 0.0007	**0.9998 ± 0.0002**	0.9996 ± 0.0001
Pythia-1.4B	0.9087 ± 0.0005	0.9022 ± 0.0022	0.9988 ± 0.0002	**0.9998 ± 0.0000**	0.9997 ± 0.0000
Pythia-2.8B	0.8972 ± 0.0008	0.8898 ± 0.0027	0.9987 ± 0.0002	0.9997 ± 0.0002	**0.9998 ± 0.0000**
Qwen3-1.7B	0.8772 ± 0.0050	0.8685 ± 0.0010	0.9985 ± 0.0009	**0.9999 ± 0.0000**	0.9996 ± 0.0000
Gemma-3-1B	0.6951 ± 0.0004	0.6517 ± 0.0012	**0.9981 ± 0.0004**	0.9570 ± 0.0047	0.9836 ± 0.0195
Park et al.					
Pythia-70M	0.8492 ± 0.0046	0.8054 ± 0.0039	0.9944 ± 0.0006	**0.9996 ± 0.0004**	0.9991 ± 0.0008
Pythia-1.4B	0.9117 ± 0.0020	0.8788 ± 0.0017	0.9988 ± 0.0002	**0.9999 ± 0.0000**	0.9992 ± 0.0000
Pythia-2.8B	0.9198 ± 0.0028	0.8925 ± 0.0016	0.9990 ± 0.0001	**0.9998 ± 0.0001**	0.9997 ± 0.0000
Qwen3-1.7B	0.9135 ± 0.0039	0.8024 ± 0.0065	0.9920 ± 0.0019	**0.9997 ± 0.0002**	0.9959 ± 0.0003
Gemma-3-1B	0.7782 ± 0.0146	0.7197 ± 0.0348	0.9573 ± 0.0034	**0.9574 ± 0.0054**	0.9502 ± 0.0007

We then compute

$$R_i^2 = \mathrm{PCC}\left(\ell_i, \widehat{\ell}_i\right)^2, \tag{71}$$

where the correlation is computed across all evaluation examples associated with concept i. The final Posterior Recovery score is

$$\text{Recovery} = \frac{1}{d_c} \sum_{i=1}^{d_c} R_i^2. \tag{72}$$

A higher score indicates that the reconstruction better preserves the marginal log-posterior coordinate associated with each concept, up to a scalar affine transformation. The results are shown in Table 7.

K Experimental Details for Out-of-Distribution Downstream Tasks

We evaluate out-of-distribution (OOD) generalization using eight classification tasks released in (Kantamneni et al., 2025). For each task, the probe is trained using examples from the original in-distribution training set and evaluated exclusively on a corresponding distribution-shifted test set. The eight tasks cover three types of distribution shift. See (Kantamneni et al., 2025) for more details.

We train an L_2-regularized logistic-regression classifier using the `LogisticRegression` implementation from scikit-learn (Pedregosa et al., 2011), with the L-BFGS solver and a maximum of 3,000 optimization iterations. We select the regularization parameter using a stratified 80/20 split of the in-distribution training data. We search over $C \in \{10^{-3}, 3\times10^{-3}, 10^{-2}, 3\times10^{-2}, 10^{-1}, 3\times10^{-1}, 1, 3, 10\}$ and select the value that maximizes validation AUC. After selection, the probe is refitted using the complete in-distribution training set and evaluated once on the OOD test set.

We report the area under the receiver operating characteristic curve (ROC-AUC). Given the predicted probability of the positive class, the ROC curve measures the trade-off between the true-positive and false-positive rates over all classification thresholds. ROC-AUC is therefore a threshold-independent measure of class separability: a value of 1 corresponds to perfect ranking.

L Ablation Studies for Sparse ConCa

Sparse ConCa contains two design choices motivated by the challenges discussed in Sec. 4. First, the module $\mathcal{R}$ in Eq. (9) regularizes the high-dimensional encoder coordinates. Its purpose is to reduce the tendency of an overcomplete autoencoder to rely on unstable or example-specific latent directions that reconstruct the input well but generalize poorly as concept coordinates. This corresponds to the overfitting issue in challenge (2). Second, Eq. (10) applies sparsity after mapping the estimated log-posterior coordinates back to the probability domain with $\exp(\hat{\mathbf{z}})$. The coefficient α controls the strength of this probability-domain sparsity constraint, addressing challenge (1). We ablate both choices to test whether the concept-extraction results depend on a particular regularization module or a narrowly tuned sparsity coefficient.

Experimental setup. We conduct the ablations with Pythia-70M and evaluate them on the 27 high-precision counterfactual concepts of Park et al. (2023), using the same testing data and evaluation protocol as in the main concept-extraction experiments. We report concept posterior alignment (MCC), concept posterior sensitivity, and concept posterior recovery (PCC^2). Unless it is the factor being ablated, the training data, dictionary width, optimizer, learning rate, number of updates, and all other training choices are fixed.

To study probability-domain sparsity, we use LayerNorm for $\mathcal{R}$ and vary

$$\alpha \in \{5\times10^{-5}, 10^{-4}, 5\times10^{-4}, 10^{-3}, 5\times10^{-3}, 10^{-2}, 5\times10^{-2}, 10^{-1}\}. \tag{73}$$

To study the module used to control overfitting, we fix $\alpha = 1 \times 10^{-4}$ and compare LayerNorm, BatchNorm, dropout, and GroupNorm as choices of $\mathcal{R}$. LayerNorm, BatchNorm, and GroupNorm control the statistics of the latent coordinates at different granularities, whereas dropout regularizes them by randomly masking coordinates during training.

Choice of regularization module. The choice of $\mathcal{R}$ materially affects whether the learned coordinates generalize as individually aligned and selective concepts. LayerNorm gives the strongest MCC and sensitivity score in this comparison, while GroupNorm remains close in MCC and dropout also performs competitively. BatchNorm is substantially weaker on both metrics. This result is consistent with the role assigned to $\mathcal{R}$ in Eq. (9): controlling each example's latent statistics with LayerNorm is more effective here than relying on batch-level statistics. Because the non-LayerNorm error bars are computed from late-training checkpoints rather than independent seeds, small differences among LayerNorm, GroupNorm, and dropout should not be interpreted as statistically conclusive. We use LayerNorm in the main experiments because it gives the strongest overall concept-level performance and is stable across three independent training runs.

Table 8: Ablations of sparse ConCa on the 27 counterfactual concepts from Park et al. (2023) using Pythia-70M. Higher is better for all three metrics.

Setting	MCC ↑	Sensitivity Score ↑	PCC^2 ↑
Choice of $\mathcal{R}$ ($\alpha = 1 \times 10^{-4}$)			
LayerNorm	0.7254 ± 0.0051	0.2265 ± 0.0355	0.99914 ± 0.00080
BatchNorm	0.6262 ± 0.0172	0.1732 ± 0.0285	0.99474 ± 0.00428
Dropout	0.7071 ± 0.0043	0.2366 ± 0.0084	0.98525 ± 0.00311
GroupNorm	0.7183 ± 0.0003	0.1724 ± 0.0237	0.99911 ± 0.00009
Sparsity strength (LayerNorm)			
$\alpha = 5 \times 10^{-5}$	0.7251 ± 0.0050	0.2296 ± 0.0326	0.99963 ± 0.00010
$\alpha = 10^{-4}$	0.7254 ± 0.0051	0.2265 ± 0.0355	0.99914 ± 0.00080
$\alpha = 5 \times 10^{-4}$	0.7219 ± 0.0055	0.2383 ± 0.0183	0.99964 ± 0.00004
$\alpha = 10^{-3}$	0.7205 ± 0.0061	0.2344 ± 0.0186	0.99964 ± 0.00007
$\alpha = 5 \times 10^{-3}$	0.7196 ± 0.0080	0.2271 ± 0.0192	0.99970 ± 0.00007
$\alpha = 10^{-2}$	0.7202 ± 0.0070	0.2277 ± 0.0055	0.99971 ± 0.00007
$\alpha = 5 \times 10^{-2}$	0.7164 ± 0.0038	0.2509 ± 0.0176	0.99955 ± 0.00015
$\alpha = 10^{-1}$	0.7166 ± 0.0036	0.2199 ± 0.0089	0.99936 ± 0.00023

Strength of probability-domain sparsity. The results are stable over more than three orders of magnitude in α. MCC obtains its largest value at $\alpha = 1 \times 10^{-4}$, while sensitivity remains comparable throughout the sweep and is highest at $\alpha = 5 \times 10^{-2}$. We use $\alpha = 1 \times 10^{-4}$ in the main experiments because it gives the best MCC and provides a favorable balance between alignment and sensitivity. Finally, affine recovery remains near ceiling for every setting, even when MCC and sensitivity differ substantially. This shows that a feature space may retain the posterior-coordinate subspace while failing to expose its individual coordinates in an aligned and selective form, and motivates reporting all three complementary metrics.