Dense SAE Latents Are Features, Not Bugs

Anonymous Author(s)
Affiliation
Address
email

Abstract

Sparse autoencoders (SAEs) are designed to extract interpretable features from language models by enforcing a sparsity constraint. Ideally, training an SAE would yield latents that are both sparse and semantically meaningful. However, many SAE latents activate frequently (i.e., are *dense*), raising concerns that they may be undesirable artifacts of the training procedure. In this work, we systematically investigate the geometry, function, and origin of dense latents and show that they are not only persistent but often reflect meaningful model representations. We first demonstrate that dense latents tend to form antipodal pairs that reconstruct specific directions in the residual stream, and that ablating their subspace suppresses the emergence of new dense features in retrained SAEs—suggesting that high density features are an intrinsic property of the residual space. We then introduce a taxonomy of dense latents, identifying classes tied to position tracking, context binding, entropy regulation, letter-specific output signals, part-of-speech, and principal component reconstruction. Finally, we analyze how these features evolve across layers, revealing a shift from structural features in early layers, to semantic features in mid layers, and finally to output-oriented signals in the last layers of the model. Our findings indicate that dense latents serve functional roles in language model computation and should not be dismissed as training noise.

1 Introduction

Sparse autoencoders (SAEs) are an unsupervised method for extracting interpretable features from language models [1, 2, 3]. They address the challenge of polysemanticity, where individual neurons activate in semantically diverse contexts that defy a single explanation [4, 5]. SAEs are trained to reconstruct the activations of a language model under a sparsity constraint applied to a bottleneck layer, ensuring that only a small subset of latents are active at a time.[1] This method effectively recovers interpretable features in a variety of models, including Claude 3 Sonnet [6] and GPT-4 [7].

Ideally, a trained SAE would yield a large set of interpretable and sparsely activating latents. In practice, however, SAEs exhibit a substantial fraction of densely activating latents, activating on 10% to 50% of tokens [8, 9]. These dense latents are challenging to interpret based solely on their activation patterns. It remains unclear whether they arise as an optimization by-product, or if they instead capture inherently dense signals present in the model's residual stream [10, 11].

In this work, we investigate several properties of dense SAE latents and the residual stream subspaces they span, uncovering evidence that these latents track meaningful residual stream information. First, we observe that when retraining an SAE on model activations with the dense latent space ablated, virtually no dense latents are learned—dense latents reflect an intrinsic property of the residual stream rather than a training artifact. We then study the geometry of dense latents and observe that they tend to form antipodal pairs, effectively reconstructing specific subspace directions.

[1] We use "latent" to refer to an entry in the SAE's sparse hidden layer.

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

We then examine the Gemma Scope suite of SAEs [12] across layers to propose a taxonomy of dense latents. We identify latents whose activations encode positional information, latents reconstructing a subspace of the residual stream linked to entropy regulation [13, 14], latents tracking high level shifts in the text, latents encoding letter-specific output signals, latents tracking parts of speech, and latents reconstructing the first residual stream principal component direction. We additionally examine how these dense latents transform across layers, finding that there is a pronounced increase in the number of dense latents just before the unembedding, as well as a shift from structural signals in early layers (e.g., position tracking) to output-oriented signals at the end. Our findings provide evidence that dense SAE latents reflect inherently dense mechanistic functions within language models.

2 Background

SAEs. Sparse autoencoders (SAEs) are trained to reconstruct a language model's activations $\mathbf{x} \in \mathbb{R}^{d_{\text{model}}}$ while imposing a sparsity constraint [15, 2]. This computation can be represented as:

$$\mathbf{f}(\mathbf{x}) := \sigma(\mathbf{W}_{\text{enc}}\mathbf{x} + \mathbf{b}_{\text{enc}}),$$
$$\hat{\mathbf{x}}(\mathbf{f}) := \mathbf{W}_{\text{dec}}\mathbf{f} + \mathbf{b}_{\text{dec}},$$

where $\mathbf{f}(\mathbf{x}) \in \mathbb{R}^{d_{\text{sae}}}$ is a sparse, non-negative vector of latents, with $d_{\text{sae}} \gg d_{\text{model}}$, and σ is a non-linear activation function. SAEs are typically trained to minimize the L2 distance between the original activation and its reconstruction $\|\mathbf{x} - \hat{\mathbf{x}}(\mathbf{f}(\mathbf{x}))\|_2^2$ while a sparsity constraint is imposed on $\mathbf{f}$ by adding a sparsity-related loss component or via specific activation functions. We denote the encoder and decoder weights of the latent at index i as $\mathbf{W}_{\text{enc}}^{(i)}$ and $\mathbf{W}_{\text{dec}}^{(i)}$, respectively. Unless noted otherwise, we use "dense" to refer to latents with an activation frequency larger than 0.1.

Experimental Setup. We focus our investigation on the Gemma Scope SAEs [12] trained on Gemma 2 2B [16], which use a JumpReLU activation function [9]. We additionally train TopK SAEs [7] on 1B tokens of the OpenWebText corpus [17] for our experiments in §3.1.[2] Activation densities for Gemma Scope latents are from Neuronpedia [18], while densities for our TopK SAEs are computed over 100M tokens from the C4 Corpus [19]. Full experimental details are in Appendix B.

3 General Properties of Dense Latents

We begin by examining structural properties of dense SAE latents, finding that they arise from a specific residual stream subspace (§3.1), and that they tend to cluster in antipodal pairs (§3.2).

3.1 Dense Latents Reflect Intrinsic Properties of the Residual Stream

To determine whether dense SAE latents arise from the training procedure or reflect an intrinsic property of the residual-stream subspace they reconstruct, we perform a targeted ablation experiment. We identify the subspace spanned by the dense latents of an SAE trained on layer 25 of Gemma 2 2B, then train a new SAE on activations in which this subspace has been zero-ablated. For comparison, we also select an equally sized set of non-dense latents and train a third SAE after ablating their subspace. We repeat this for two dictionary sizes ($d_{\text{sae}} = 16384$ and 32768).

Figure 1a shows the resulting distributions of latent activation densities. In both dictionary sizes, ablating the dense-latent subspace (teal) yields much fewer high-density latents than the original SAE (blue) and the non-dense ablation (orange). This result implies that densely activating latents are not mere training artifacts but instead track a dense residual-stream subspace whose presence drives the emergence of dense latents. As additional evidence that dense latents are not training artifacts, in Appendix A.2 we show that longer training does not reduce the number of dense latents.

3.2 Dense Latents Cluster in Antipodal Pairs

We now examine the geometry of dense latents and observe that they tend to form antipodal pairs. That is, as shown in Figure 1b, there exist many pairs of two dense latents that have nearly opposite

[2] We choose TopK for its reliable training and competitive reconstruction–sparsity trade-off.

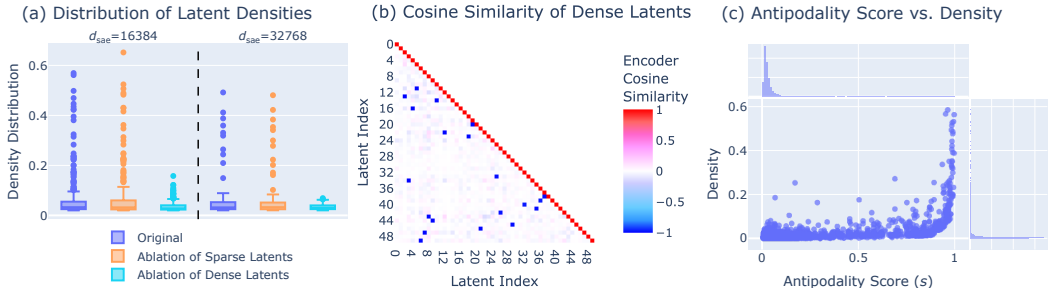

Figure 1: **General Properties of Dense SAE Latents.** (a) Ablating the dense-latent subspace (teal) reduces high-density latents compared to the original (blue) and sparse-latent ablations (orange). (b) Encoder cosine similarity between the top 50 latents with highest density. (c) Dense latents exhibit high antipodality score: they form pairs that reconstruct specific residual stream directions.

decoder vectors (we find a similar result for encoder vectors). This suggests that the SAE allocates two latents in the dictionary to represent a 1-dimensional line.

To quantify whether this phenomenon is specific to dense latents, we introduce an antipodality score s_i for a latent i. We first compute the pairwise cosine similarities between the latent's weights (both encoder and decoder) and those of all other latents. Then, we compute the maximum product of encoder and decoder cosine similarity across all pairs (i, j) for all $i \neq j$. Formally, we have

$$s_i := \max_{j \neq i} \left(\text{sim}\left(\mathbf{W}_{\text{enc}}^{(i)}, \mathbf{W}_{\text{enc}}^{(j)}\right) \cdot \text{sim}\left(\mathbf{W}_{\text{dec}}^{(i)}, \mathbf{W}_{\text{dec}}^{(j)}\right) \right), \tag{1}$$

where $\text{sim}(u, v)$ denotes the cosine similarity between vectors u and v. This score reflects the extent to which latent i forms an antipodal pairing with another latent: high values of s_i indicate that there is another latent j with both encoder and decoder weights nearly opposite in direction to those of i.[3]

As shown in Figure 1c, s_i and the activation density of latent i are strongly positively correlated. The majority of dense latents–particularly those with an activation frequency exceeding 0.3–exhibit pairwise scores greater than 0.9, supporting our conclusions above. We provide density-antipodality visualizations for additional SAEs in Appendix A.1, showing that this trend holds consistently across SAE architectures (JumpReLU and TopK), models (GPT-2 and Gemma), and layers.

4 Taxonomy

Having established that dense latents are persistent and geometrically structured, we now investigate their interpretability. We identify classes of dense latents based on the model signals they represent:

- **Position latents** (§4.1) fire based on token position relative to structural boundaries (start of sentence, paragraph or context) and appear early in the network.
- **Context-binding latents** (§4.2) represent context-dependent semantic content and exhibit coherent chunk-level activations, potentially representing high-level ideas within the context.
- **Nullspace latents** (§4.3) track components of the residual stream that have minimal impact on next token prediction. They instead regulate prediction entropy.
- **Alphabet latents** (§4.4) promote broad sets of tokens sharing an initial character.
- **Meaningful-word latents** (§4.5) have activations related to the token part-of-speech tag.
- **PCA latents** (§4.6) lie almost completely within the first PCA components of the activation space.

4.1 Position Latents

We first identify a class of dense latents whose activations track the current token's position relative to specific text boundaries. **Context-tracking** latents track token position w.r.t. the BOS token,

[3] Although high values of s could be produced by two nearly identical latents, retaining such a pair would be redundant–a scenario we do not observe. Evidence for this is provided in Appendix A.4.

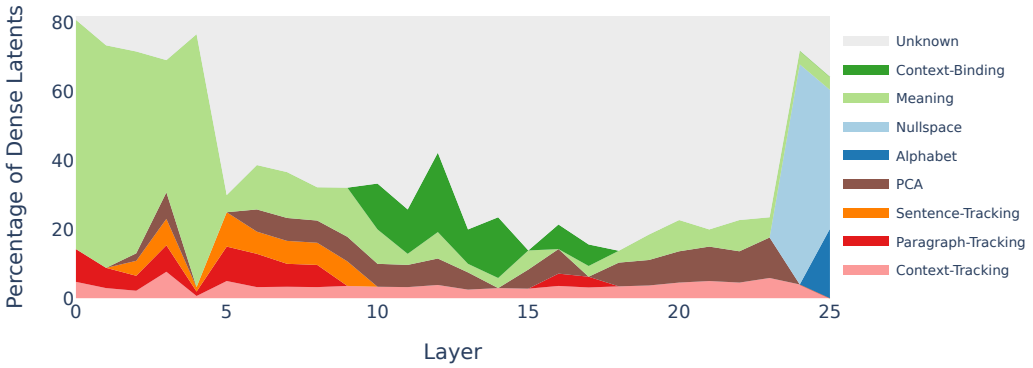

Figure 2: **An overview of our taxonomy of dense latents, for every layer.** See Appendix E.1 for how we created this plot.

paragraph-tracking latents track token position w.r.t. a paragraph start, and **sentence-tracking** latents track token position w.r.t. a sentence beginning. Context-position latents are similar to "position neurons" from prior work [20]; the other categories are to the best of our knowledge novel.

To find these latents systematically, we use Spearman's rank correlation coefficent ρ. For each dense latent, we capture the projections[4] of the residual stream activations onto its decoder vector for 5000 1024-token-long contexts. We find ρ between this projection and the distance from the last period, the last newline, and the beginning of the input. These boundaries act as proxies for "beginning of sentence", "beginning of paragraph" and "beginning of context", respectively.

Figure 2 shows the resulting trends: sentence-tracking and paragraph-tracking latents are prominent before layer 10, while context-position tracking latents are present throughout the model. Figure 13 shows ρ for all latents across layers. We can clearly see groups of outlier latents for each category, and thus classify latents as belonging to that category if $|\rho| > 0.4$. Indeed, examples in Appendix E.2 confirm that the identified outlier latents have position-tracking behavior. Notably, Appendix E.2 also shows that paragraph-tracking latents are agnostic to artificially adding formatting newlines, suggesting that this direction in the model tracks true semantic paragraph breaks. Thus, our "distance to new line objective" is just a proxy. We also note that latents with high ρ with periods also have high ρ with newlines, since newlines and periods are correlated in text. In Figure 15, we thus show the ρ for sentence-tracking vs. paragraph-tracking across all dense latents.

At a higher level, it makes sense that the model represents these features in a dense way: positional information is always relevant to the model's predictions (e.g., it must track how far it is in a sentence to correctly predict a period), so the model might store this representation in a consistent direction in every hidden state, which is then learned by the SAE.

4.2 Context-Binding Latents

We next identify a class of dense latents that encode different semantic concepts depending on context. Unlike interpretable sparse SAE latents typically associated with fixed meanings, such as the "Golden Gate Bridge" feature in Claude [6], these dense latents appear to *bind* to the main ideas of the context.

We first observe that some dense latents, particularly in middle layers, activate on long consecutive "chunks" of tokens.[5] Examining the activations of such latents and attempting to explain them with an LLM (see Appendix E.5), we notice empirically that such latents fire on highly specific concepts *within a context*, but the concepts *vary across contexts*. One possible interpretation is that these latents represent general but abstract, difficult-to-interpret properties. However, we also observe that within an antipodal pair, the active latent often switches when the main topic or entity in the text changes

[4]We use the projection of the residual stream rather than the JumpRELU activations of these latents since we hypothesize that the *direction* itself encodes the positional information, regardless of whether the magnitude exceeds the learned JumpRELU threshold.

[5]While positional latents also exhibit consecutive activations, here we refer to non-positional latents whose activations cannot be explained by position alone.

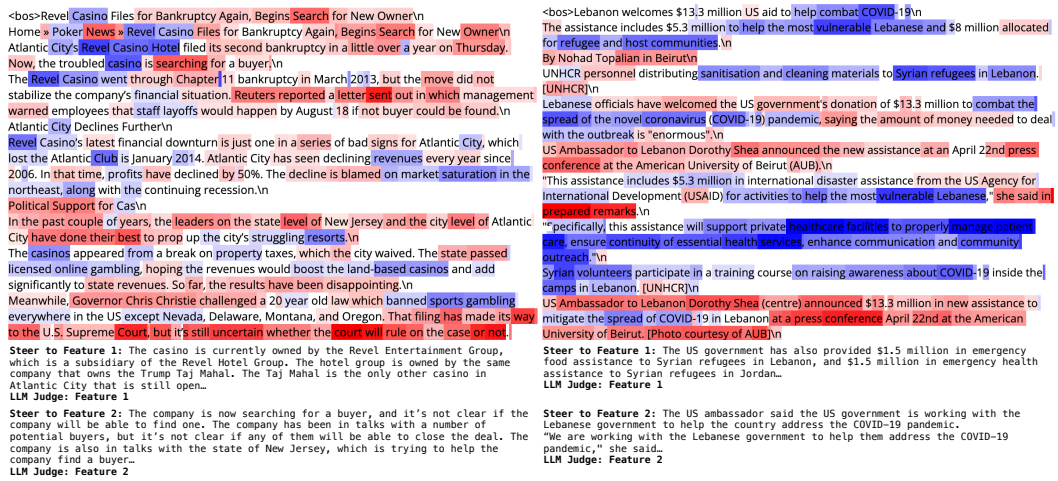

Figure 3: **Context-Binding Latents.** Activation patterns of layer 12 antipodal pair 7541 (blue, feature 1) and 2009 (red, feature 2). In the first context, they seem to be tracking "casino facts" vs "looking for a buyer", while in the second context, they seem to be tracking "healthcare" vs "press conference". Their corresponding completions are in line with the concepts they activated on.

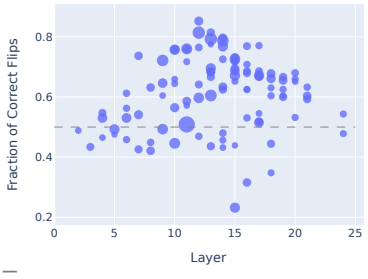

Figure 4: **Fraction of correct flips when steering**, for all latent pairs that have at least one latent $f > 0.2$, and ≥ 40 flips. Points are sized by number of flips.

Layer	Latent Pair	In-context	Out-of-context
12	(14906, 14599)	0.051	0.717
12	(2291, 13295)	0.028	0.760
12	(7541, 2009)	0.043	0.711
13	(3517, 46)	0.036	0.742
13	(15275, 11449)	0.029	0.704
13	(12613, 7655)	0.028	0.531
14	(11575, 2411)	0.047	0.798
14	(8515, 15297)	0.041	0.603
14	(6699, 1802)	0.037	0.678
16	(2889, 8811)	0.024	0.665
17	(10495, 491)	0.051	0.669

Table 1: **Fraction of "unclear" judgements** using in-context examples versus out-of-context examples, for the highest-scoring latents by flips.

(Figure 3, Appendix E.3). This raises the hypothesis that such directions act as a "registers" in the residual stream for tracking the active concept, rather than simply representing generic properties.

We thus perform a steering experiment to find the causal effect of these directions. For each antipodal pair (F1, F2), we prompt Gemma 2 2B with input text from the RedPajama dataset [21] and generate completions without steering, steering to F1, and steering to F2. An LLM judge [22] is then asked whether each completion is more in line with activating examples (from the input context) of F1 or F2, or unclear. Further details of the methodology are in Appendix E.4.

Since the unsteered generation may already favor F1 or F2, we quantify steering success by the fraction of *flips* from the unsteered judgement that align correctly with the steering direction. For several mid-layer latent pairs, steering reliably shifts completions toward the specific concept previously associated with the latent *in that context*. However, when judged against out-of-context examples, the rate of unclear judgements rises sharply. While difficult to rule out the possibility that these directions encode "general uninterpretable" features, the specificity of the steered generation in bringing up context-related ideas (Figure 3) suggests that these latents could bind to concepts in a context-dependent, rather than globally consistent, way.

Previous works have uncovered "binding mechanisms" that help the model keep track of in-context associations between entities [23, 24]. While our findings do not directly prove such a mechanism, they raise the possibility that dense subspaces may play a similar functional role, distinguishing

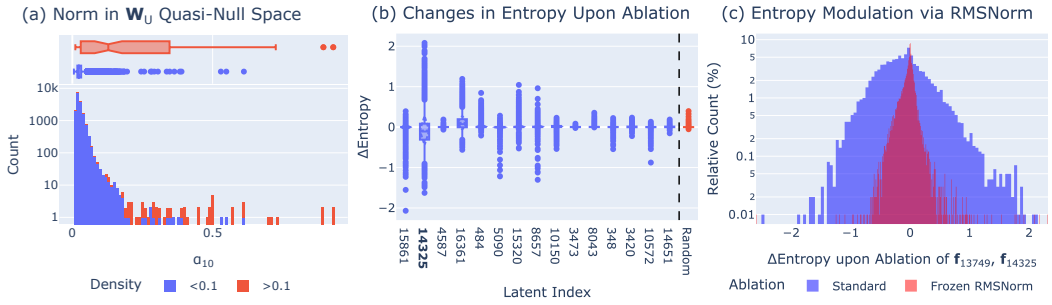

Figure 5: **Nullspace Latents.** (a) A small fraction of latents concentrate norm in the final 10 singular directions of $\mathbf{W}_U$, with high-density latents overrepresented in this group. (b) A pair of such latents correlates strongly with model output entropy. (c) Ablating this pair lowers entropy; the effect substantially decreases when RMSNorm scaling is frozen.

the currently active semantic concept. Further work could explore the circuits [25] involving such subspaces, and challenge the assumption of globally monosemantic directions.

4.3 Nullspace Latents

Previous work has identified a $\mathbf{W}_U$ *quasi-nullspace*–the subspace spanned by the last singular vectors of the unembedding matrix $\mathbf{W}_U$–which accounts for a substantial portion of the residual stream's norm, yet has little direct impact on next-token prediction [14]. Since this subspace carries high norm, we hypothesize that some dense SAE latents are allocated specifically to reconstruct it.

To test this, we compute the singular value decomposition $\mathbf{W}_U = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$. Then, we study the composition of an SAE latent i's encoder weight with the space spanned by the last k left singular vectors $\mathbf{U}_{-k}, \ldots, \mathbf{U}_{-1}$ of $\mathbf{W}_U$ by computing the fraction ρ_k of the norm of its encoder weight $\mathbf{W}_{\text{enc}}^{(i)}$ that lies in this subspace:

$$\alpha_k = \frac{\sum_{j=1}^{k} \mathbf{U}_{-j}^T \mathbf{W}_{\text{enc}}^{(i)}}{\|\mathbf{W}_{\text{enc}}^{(i)}\|}. \tag{2}$$

A histogram of α_{10} for the SAE trained at layer 25 of Gemma 2 2B (Figure 5a) shows that 99.6% of latents have $\alpha_{10} < 0.2$. We designate those with $\alpha_{10} > 0.2$ as *nullspace-aligned*. Interestingly, 75% of them are high-density, and account for 40% of the high-density latents in the SAE.

Unlike other dense latents, nullspace-aligned latents are hard to interpret via their token-level activation patterns. Additionally, the tokens they promote are typically uninterpretable "under-trained" tokens [26]. Motivated by prior work linking the $\mathbf{W}_U$ nullspace to an RMSNorm-based [27] entropy regulation mechanism [13], we investigate whether these latents encode this internal computation.

To test whether these latents causally influence output entropy, we ablate the residual stream along each latent's decoder direction by setting its value to the corresponding decoder bias, thereby removing information in that direction. We then measure the change in per-token entropy of the model's output distribution. Figure 5b reports the entropy change for all latents with $\alpha_{10} > 0.3$ (one per antipodal pair to avoid redundancy), compared to a control group of 50 randomly selected latents.[6]

We find that some nullspace latents produce much larger entropy shifts than the random baseline, indicating that they encode signals relevant to entropy modulation. In particular, latent 14325 has a disproportionate impact on output entropy. To test whether this signal is used by the model in conjunction with RMSNorm scaling (as in [13]), we repeat the ablation while freezing the RMSNorm scaling coefficient. Figure 5c shows that the entropy change diminishes under this intervention, suggesting that the model uses this direction to modulate entropy via RMSNorm. Furthermore, Figure 6 shows that the combined activation of the antipodal pair formed by latents 13748 and 14325 is strongly correlated with output entropy, further supporting this interpretation.

While these results highlight the functional role of specific nullspace latents in entropy regulation, not all latents in this subspace behave similarly. Some exhibit negligible impact on entropy when ablated.

[6]The entropy changes for the random latents are aggregated into a single boxplot.

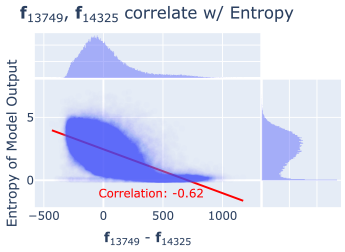

Figure 6: **Entropy Correlation.** A pair $\mathbf{W}_U$ nullspace-aligned correlates strongly with model output entropy.

Index	Letter	Density	Metric	Top Tokens
15287	R	0.16	0.98	_RI, _rb, getR, _ri, _r, _RS, R, _RR
13531	M	0.15	0.97	_MM, _m, MM, _mM, _mm, _mf, _ms, mM
30	T	0.16	0.99	_TT, _TC, TT, TC, _tc, _TG, _TS, _TD
1761	D	0.14	0.98	_DD, _D, _DS, _DP, _DT, DD, DP, DS, _Ds
7342	I	0.13	0.91	IB, i, IC, İ, IE, IH, IP, _IW, IR, IW
2651	U	0.11	0.93	_UA, U, _UT, UU, _U, _UF, _UD, UE, UA
4664	C	0.14	0.93	_getC, _CC, getC, _c, setC, CC, Cs, _Cs
357	B(+R)	0.006	0.91	_BR, _Br, Br, BR, _Bra, _br, Bra, br
12114	S(+L)	0.006	0.95	_SL, SL, _sl, _Sl, sl, Sl, _Slide
14857	C(+U)	0.006	0.91	_Cur, _cur, Cur, _CUR, cur, CUR, _Kur

Table 2: **Examples of Alphabet Latents.** Latents from layer 25 of Gemma 2 2B that promote or suppress tokens sharing an initial letter. "Metric" is the fraction of top 100 affected tokens starting with that letter.

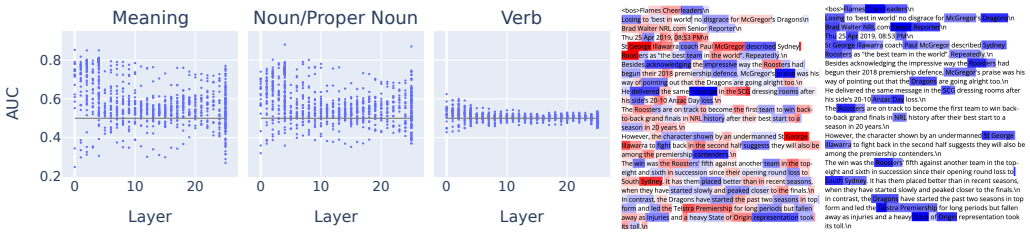

Figure 7: **Meaningful-Word Latents.** (Left) AUCs of predicting feature firing, from whether the POS tag is within the specific category. "Meaning word" and "noun/propernoun" are good predictors, while other categories like "verb" are less predictive. (Middle) Example of L2: pair 15089 (blue), 13092 (red) firing pattens on a document, where 15089 fires on "meaning-heavy" words while 13092 fires on proper nouns and functional words (the, in, a). (Right) Example of L3: 7507 firing patterns, where it fires selectively on proper nouns.

We speculate that these may track different internal signals–one such candidate is the attention sink signal, which has also been associated with the $\mathbf{W}_U$ nullspace [14]. Overall, these experiments provide mechanistic evidence that nullspace latents correspond to internal model computations.

4.4 Alphabet Latents

We identify a class of dense latents that selectively boost or suppress large sets of tokens sharing the same initial letter. Unlike prior work that linked latents to the *current* token's first letter [28], these instead relate to the *next* token's initial character.

To discover these latents systematically, we examine each latent's top 100 positive and negative logit contributions by projecting its decoder weights onto the vocabulary space. Then, we collect the corresponding tokens, and select latents where either set contains at least 90% of tokens starting with the same character (excluding the space character "_"). At layer 25, this procedure yields 114 such latents, of which 21 have activation density >0.1, accounting for 20% of all dense latents. These latents span a range of antipodality scores and activation densities, but notably appear as high-density features only at the model's final layer. We provide some examples from this layer in Table 2.

Interestingly, we observe multiple latents for each letter, varying in specificity: some target a broad set of short tokens sharing only the first letter (e.g., "b" or "c"), while others focus on longer tokens sharing a multi-letter prefix (e.g., "br" or "cu"). We attribute this granularity to feature splitting [1] possibly driven by n-gram frequency, which yields latents with differing activation densities. These latents illustrate how SAEs dedicate dense units to encode output-specific signals related to next-token lexical structure.

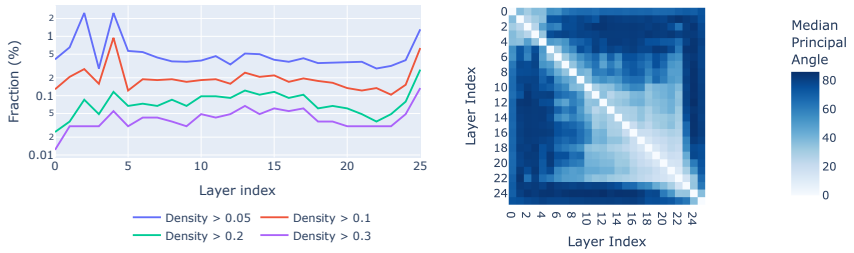

Figure 8: **Layer-wise Dynamics of Dense Latents.** (a) Fraction of dense latents (at various density thresholds) across residual stream SAEs at different layers of Gemma 2 2B. (b) Median principal angles between dense-latent subspaces, showing a shift in subspace structure from early to late layers.

4.5 Meaningful-Word Latents

The next class of latents that we investigate are those whose firing can be well predicted by the part-of-speech (POS) tag of the token. We create a reduced set of high-level tags from the Brown Corpus [29] by combining similar tags (e.g., combining plural and singular forms of nouns),[7] and capture dense latent activations on 10k sentences ($\approx$ 200k tokens) from the corpus. Then, for each latent, we calculate the AUC-ROC of predicting the binary latent activations given the binary vector of whether a token is within the high-level POS category. Intuitively, this AUC reflects how well the interpretable linguistic category *predicts* the latent.

We find that even these high-level groupings are not enough to achieve a high AUC (Figures 7 and 18), and propose a further grouping of these tags into "meaningful words", where a token is considered a "meaningful word" if it is one of {nouns, proper nouns, verbs, adjectives, adverbs}. The resulting binary-binary predictor has a decent AUC (Figure 7) of ≈ 0.8 for many dense latents in early layers, suggesting that the model contains a dense subspace tracking the presence of these meaningful words.

4.6 PCA Latents

Since the top principal components (PCs) are a large fraction of the variance of the activations, one might expect an SAE to learn dense latents that simply reconstruct this subspace. However, we find that this hypothesis is only partly the case: as shown in Figure 19, an antipodal pair of latents consistently reconstruct most of the first PC (cosine similarity > 0.75), but other latents do *not* have a large norm percentage in the top PC, even up to the top 5 PC components. The top PC-aligned latents are generally not immediately interpretable and do not fall into any of our classes above. Interestingly, decreasing or increasing the SAE L_0 and dictionary size does not get rid of PC-aligned latents nor result in significantly more of them (Figure 20).

4.7 Layer-wise Dynamics

As noted in the taxonomy of dense latents above, and visualized in Figure 2, each class of dense latents is found in specific layer ranges. Dense latents in early layers have more token-dependent activations and track positional information, those in middle layers represent more conceptual directions, and those in the final layers are mostly mechanisms that the model uses to control its output. Inspired by these observations, in this section we further examine layer-wise characteristics of dense latents.

Number of Dense Latents. First, we study how the number of dense latents changes across different layers of the model. Figure 8a illustrates the fraction of latents exceeding density thresholds of 0.05, 0.1, 0.2, and 0.3 at each layer. In the early layers (0-4), we observe transient spikes in latents just above the 0.05 and 0.1 thresholds. These latents are largely the part-of-speech related latents in §4.5. The absence of similar spikes at the 0.2 and 0.3 thresholds suggest that these early fluctuations arise from SAE training variability rather than fundamental differences in the information encoded at different points of the model's residual stream. Across the middle layers (5–23), the fraction of dense latents is remarkably stable for all thresholds. Finally, the model's last two layers exhibit an increase in the number of dense latents, indicating a final emergence of dense features prior to unembedding.

[7] See Table 4 in Appendix E.6 for our full mapping.

Consistency of the Dense Subspace. We next ask whether the subspace spanned by dense latents remains stable across layers or varies over the model. For each pair of layers, we compute the principal angles between the subspaces defined by latents with density > 0.2, then take the median angle as a summary statistic: values near $0°$ indicate largely overlapping subspaces, while values near $90°$ indicate dissimilarity. Figure 8c visualizes these median angles for every layer pair of Gemma 2 2B.[8] Three clusters emerge. Layers 0-4 share a common dense subspace (low angles). This shifts in the middle of the model (layers 10–22), where a new stable subspace persists (mutually low angles). Finally, the last few layers exhibit a pronounced change (large angles relative to earlier layers), consistent with the rise of alphabet and nullspace latents before the unembedding.

5 Related Work

Sparse Autoencoders. Transformer models are thought to represent features as linear directions in activation space [30, 31, 32, 33, 34, 35], with many more features than neurons, leading to *superposition* [36, 5]. Early work explored sparse dictionary learning to interpret these representations [37, 38, 39, 40]. More recently, sparse autoencoders (SAEs; 41) have emerged as a scalable and effective implementation of sparse dictionary learning for transformer-based models [15, 1, 2, 42, 9, 3, 43] that can recover meaningful and causally important features [6, 7, 25].

Interpreting SAE Latents. As SAEs have gained traction, recent work has focused on interpreting the meaning of their latent features [28, 44]. Building on the neuron interpretation methodology in [45], several recent works interpret SAE latents systematically with LLMs [6, 46] or automated interpretation strategies [47]. A recurring observation across multiple studies are *dense* latents, which activate on more than 10% or even 50% of tokens [8, 9]. Whether these latents reflect meaningful internal computations or arise as undesirable artifacts was up until our work an open question [10, 11].

Dense Language Model Representations. A few prior works identify dense LLM representations: two studies [20, 48] identify positional features in LLMs, and another study [13] finds entropy directions that are always active. Both of these feature types are in our taxonomy of SAE latents.

6 Discussion, Limitations & Conclusion

Our work shows that dense SAE latents discover intrinsically dense features in the underlying language model representations. This challenges recent efforts that aim to remove dense latents with add-hoc penalties in the SAE loss function [11]. Our results motivate future feature-extraction mechanisms that are able to find features that are not necessarily sparse. For example, such techniques might include SAE designs that allocate autoencoder capacity for representing dense subspaces, approaches that optimize circuit sparsity, or techniques like APD [49] that focus on parameter sparsity.

Limitations. Although our work identifies some classes of dense latents, we do not claim that all dense latents encode interpretable or meaningful signals. We hypothesize that some dense latents are a noisy aggregation of sparse features rather than a "true" dense feature, and distinguishing between these remains an open challenge. Moreover, dense latents may learn a basis that spans *but does not align with* the set of true dense model representations, since dense latents co-occur extremely frequently, and a linear combination of the "true" basis works for reconstruction too.

Despite consistently observing the antipodality trend across both TopK and JumpReLU SAEs and across models (Gemma 2 2B and GPT-2 Small), our interpretability analysis primarily focuses on JumpReLU SAEs trained on Gemma-2-2B, using a single dictionary size and sparsity constraint per layer. Future work could broaden analysis to more models, SAE architectures, and SAE sparsities.

Most notably, we have explained less than half of dense SAE features. We view understanding the rest of these latents as exciting future work that could provide insight into frequently-active, fundamental mechanisms and representations in language models.

[8] We find that using a slightly higher density threshold (> 0.2) makes the subspace similarity pattern more pronounced. The same plot with a lower threshold (> 0.1) is shown in Appendix A.3, showing the same clustering trend but with reduced overall similarity.

References

[1] 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.

[2] 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*, 2024.

[3] Connor Kissane, Robert Krzyzanowski, Joseph Isaac Bloom, Arthur Conmy, and Neel Nanda. Interpreting attention layer outputs with sparse autoencoders. In *ICML 2024 Workshop on Mechanistic Interpretability*, 2024.

[4] Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. Feature visualization. *Distill*, 2017.

[5] 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.

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

[7] 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.

[8] Hoagy Cunningham and Tom Conerly. Comparing topk and gated saes to standard saes, 2024.

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

[10] Brian Chen and Josh Batson. Interpretable dense features, 2025.

[11] Senthooran Rajamanoharan, Callum McDougall, and Lewis Smith. Removing high frequency latents from jumprelu saes, 2025.

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

[13] Alessandro Stolfo, Ben Peng Wu, Wes Gurnee, Yonatan Belinkov, Xingyi Song, Mrinmaya Sachan, and Neel Nanda. Confidence regulation neurons in language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[14] Nicola Cancedda. Spectral filters, dark signals, and attention sinks. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4792–4808, Bangkok, Thailand, August 2024. Association for Computational Linguistics.

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

[16] Gemma Team. Gemma 2: Improving open language models at a practical size, 2024.

[17] Aaron Gokaslan and Vanya Cohen. Openwebtext corpus, 2019.

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

[19] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.

[20] Wes Gurnee, Theo Horsley, Zifan Carl Guo, Tara Rezaei Kheirkhah, Qinyi Sun, Will Hathaway, Neel Nanda, and Dimitris Bertsimas. Universal neurons in gpt2 language models, 2024.

[21] Maurice Weber, Daniel Y. Fu, Quentin Anthony, Yonatan Oren, Shane Adams, Anton Alexandrov, Xiaozhong Lyu, Huu Nguyen, Xiaozhe Yao, Virginia Adams, Ben Athiwaratkun, Rahul Chalamala, Kezhen Chen, Max Ryabinin, Tri Dao, Percy Liang, Christopher Ré, Irina Rish, and Ce Zhang. Redpajama: an open dataset for training large language models. *NeurIPS Datasets and Benchmarks Track*, 2024.

[22] Google Cloud. Gemini 2.5 flash | generative ai on vertex ai, 2025.

[23] Jiahai Feng and Jacob Steinhardt. How do language models bind entities in context?, 2024.

[24] Jiahai Feng, Stuart Russell, and Jacob Steinhardt. Monitoring latent world states in language models with propositional probes, 2024.

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

[26] Sander Land and Max Bartolo. Fishing for magikarp: Automatically detecting under-trained tokens in large language models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 11631–11646, Miami, Florida, USA, November 2024. Association for Computational Linguistics.

[27] Biao Zhang and Rico Sennrich. Root mean square layer normalization. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[28] David Chanin, James Wilken-Smith, Tomáš Dulka, Hardik Bhatnagar, and Joseph Bloom. A is for absorption: Studying feature splitting and absorption in sparse autoencoders, 2024.

[29] W. N. Francis and H. Kučera. Brown corpus manual: A standard corpus of present-day edited american english, for use with digital computers. Technical report, Department of Linguistics, Brown University, Providence, Rhode Island, 1979.

[30] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In C.J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013.

[31] Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

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

[33] Neel Nanda, Andrew Lee, and Martin Wattenberg. Emergent linear representations in world models of self-supervised sequence models. In Yonatan Belinkov, Sophie Hao, Jaap Jumelet, Najoung Kim, Arya McCarthy, and Hosein Mohebbi, editors, *Proceedings of the 6th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 16–30, Singapore, December 2023. Association for Computational Linguistics.

[34] Kiho Park, Yo Joong Choe, and Victor Veitch. The linear representation hypothesis and the geometry of large language models. In Salakhutdinov, Ruslan and Kolter, Zico and Heller, Katherine and Weller, Adrian and Oliver, Nuria and Scarlett, Jonathan and Berkenkamp, Felix, editor, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 39643–39666. PMLR, 21–27 Jul 2024.

[35] Christopher Olah. What is a linear representation? what is a multidimensional feature? *Transformer Circuits Thread*, 2024.

[36] Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. Zoom in: An introduction to circuits. *Distill*, 2020.

[37] Bruno A. Olshausen and David J. Field. Sparse coding with an overcomplete basis set: A strategy employed by v1? *Vision Research*, 37(23):3311–3325, 1997.

[38] Manaal Faruqui, Yulia Tsvetkov, Dani Yogatama, Chris Dyer, and Noah A. Smith. Sparse overcomplete word vector representations. In Chengqing Zong and Michael Strube, editors, *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1491–1500, Beijing, China, July 2015. Association for Computational Linguistics.

[39] Sanjeev Arora, Yuanzhi Li, Yingyu Liang, Tengyu Ma, and Andrej Risteski. Linear algebraic structure of word senses, with applications to polysemy. *Transactions of the Association for Computational Linguistics*, 6:483–495, 2018.

[40] Juexiao Zhang, Yubei Chen, Brian Cheung, and Bruno A Olshausen. Word embedding visualization via dictionary learning, 2021.

[41] Andrew Ng et al. Sparse autoencoder. *CS294A Lecture notes*, 72(2011):1–19, 2011.

[42] Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Tom Lieberum, Vikrant Varma, Janos Kramar, Rohin Shah, and Neel Nanda. Improving sparse decomposition of language model activations with gated sparse autoencoders. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[43] Bart Bussmann, Noa Nabeshima, Adam Karvonen, and Neel Nanda. Learning multi-level features with matryoshka sparse autoencoders, 2025.

[44] Patrick Leask, Bart Bussmann, Michael T Pearce, Joseph Isaac Bloom, Curt Tigges, Noura Al Moubayed, Lee Sharkey, and Neel Nanda. Sparse autoencoders do not find canonical units of analysis. In *The Thirteenth International Conference on Learning Representations*, 2025.

[45] Steven Bills, Nick Cammarata, Dan Mossing, Henk Tillman, Leo Gao, Gabriel Goh, Ilya Sutskever, Jan Leike, Jeff Wu, and William Saunders. Language models can explain neurons in language models, 2023.

[46] Gonçalo Paulo, Alex Mallen, Caden Juang, and Nora Belrose. Automatically interpreting millions of features in large language models, 2024.

[47] Dmitrii Kharlapenko, Stepan Shabalin, Neel Nanda, and Arthur Conmy. Self-explaining sae features, 2024.

[48] Bilal Chughtai and Yeu-Tong Lau. Understanding positional features in layer 0 saes, July 2024. LessWrong post.

[49] Dan Braun, Lucius Bushnaq, Stefan Heimersheim, Jake Mendel, and Lee Sharkey. Interpretability in parameter space: Minimizing mechanistic description length with attribution-based parameter decomposition. *arXiv preprint arXiv:2501.14926*, 2025.

[50] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[51] Neel Nanda and Joseph Bloom. Transformerlens. `https://github.com/TransformerLensOrg/TransformerLens`, 2022.

[52] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, sep 2020.

[53] Wes McKinney. Data Structures for Statistical Computing in Python. In Stéfan van der Walt and Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 56 – 61, 2010.

[54] Plotly Technologies Inc. Collaborative data science, 2015.

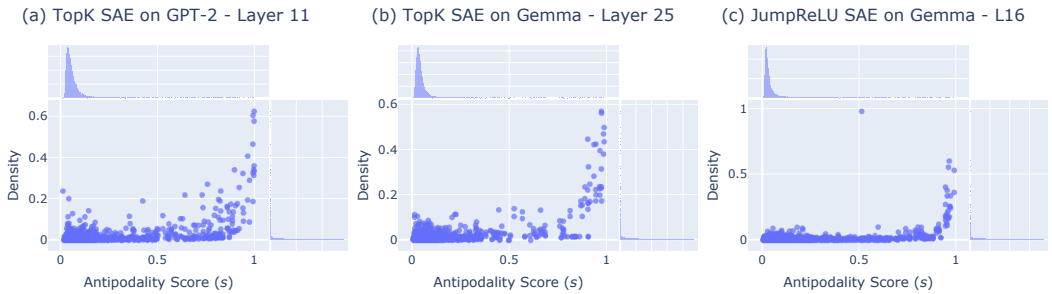

Figure 9: **Additional Antipodality Plots.** Antipodality scores vs. activation density for (a) TopK SAE on GPT-2 (Layer 11), (b) TopK SAE on Gemma 2 2B (Layer 25), and (c) JumpReLU SAE on Gemma 2 2B (Layer 16). Across all configurations, dense latents tend to have high antipodality scores.

A Additional Results

A.1 Antipodal Pairing in Different SAEs

Figure 9, we report antipodality scores (computed as in Eq. (1)) for dense latents in three additional SAEs: two TopK SAEs that we trained on the residual streams of GPT-2 (layer 11) and Gemma 2 2B (layer 25), and a JumpReLU SAE from the Gemma Scope suite trained on an earlier layer (16). In all cases, we observe the same trend highlighted in Figure 1c: high-density latents cluster at high antipodality scores, forming near-antipodal pairs that reconstruct specific directions in residual space.

A.2 Dense Latents During Training

In Figure 10, we visualize the number of dense latents (activation frequency > 0.1) over training steps for each SAE configuration in our ablation experiment described in §3.1. All curves converge within the first $\sim$100k steps and remain stable throughout training. This early plateau suggests that dense latents are not a product of late-stage optimization noise, but rather emerge early and persist, indicating that they reflect consistent structure in the residual stream rather than transient artifacts.

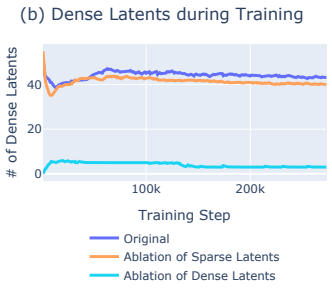

Figure 10: **Dense Latents During Training.** Dense latent counts stabilize early in training.

A.3 Angles Between Residual Stream Subspaces

In Figure 11, we provide further analysis of the evolution of dense latent subspaces across layers. Panel (a) shows the median principal angle between the subspaces spanned by latents with density > 0.1 at each pair of layers in Gemma 2 2B. These results follow the trend observed in Figure 8c (based on a > 0.2 cutoff), revealing distinct subspace clusters in the early, middle, and late layers. However, the overall similarity between subspaces is lower here, reflecting the greater variability introduced by including moderately dense latents (density 0.1-0.2).

For comparison, panel (b) reports the same metric computed on subspaces spanned by 100 randomly selected non-dense latents per layer. As expected, these subspaces exhibit minimal overlap, with median principal angles near 90° across all layer pairs, confirming that the structure observed in the dense-latent subspaces is nontrivial.

A.4 Pairwise Similarity Between Latents' Weights

In Figure 11c, we report for each latent i, the maximum-magnitude cosine similarity of its encoder and decoder weights with any other latent j. In particular, we show $\mathrm{sim}(\mathbf{W}_{\mathrm{enc}}^{(i)}, \mathbf{W}_{\mathrm{enc}}^{(j)})$ and $\mathrm{sim}(\mathbf{W}_{\mathrm{dec}}^{(i)}, \mathbf{W}_{\mathrm{dec}}^{(k)})$, where $j = \arg\max_{l \neq i}(|\,\mathrm{sim}(\mathbf{W}_{\mathrm{enc}}^{(i)}, \mathbf{W}_{\mathrm{enc}}^{(l)})|)$ and $k =$

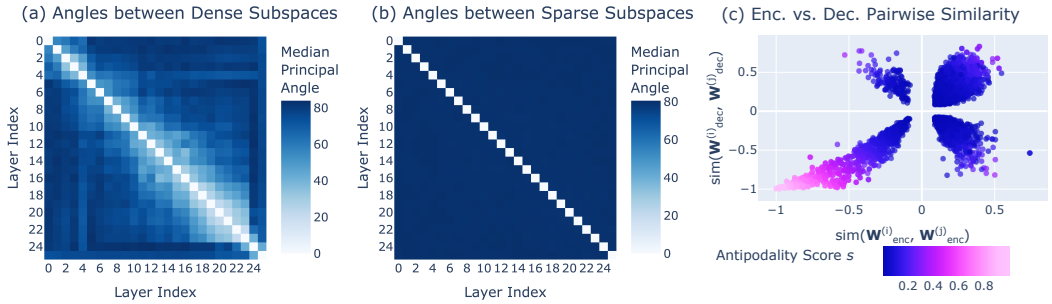

Figure 11: **Additional Analyses.** (a) Median principal angles between dense-latent subspaces (density>0.1) across layers. (b) Principal angles between randomly selected non-dense latent subspaces. (c) High antipodality score occurs when encoder and decoder weights are nearly opposite.

$\arg\max_{l \neq i}(|\operatorname{sim}(\mathbf{W}_{\text{dec}}^{(i)}, \mathbf{W}_{\text{dec}}^{(l)})|)$. We find that the antipodality score s approaches 1 only when both encoder and decoder similarities are close to -1.

A.5 Similarity with SAE Bias

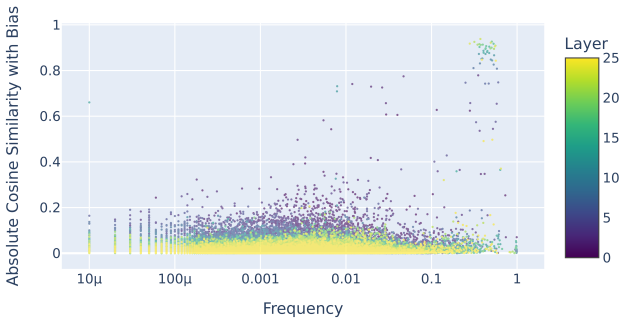

Figure 12: Plot of absolute cosine similarity of all SAE decoder vectors at all layers with that layer's decoder bias. We observe a group of dense latents in the upper right corner that have high frequency and align with the bias.

B Experimental Details

For the experiment in §3.1, we trained TopK SAEs [7] on the residual stream activations at layer 25 of Gemma 2 2B using 1 billion tokens from the OpenWebText corpus [17]. Training followed the default configuration of the `Sparsify` library,[9] and experiment tracking was conducted using Weights & Biases.[10] The ablation experiment on nullspace latents described in §4.3 was performed on a 10k-token subset of the C4 corpus [19]. Analyses throughout the paper were conducted using the Gemma Scope SAEs [12] with 16k latents trained on the residual stream of Gemma 2 2B. All experiments were implemented in `PyTorch` [50], with model inspection tools from the `TransformerLens` library [51]. Data processing used `NumPy` [52] and `Pandas` [53], and figures were generated with `Plotly` [54].

C Compute Resources Used

We expect the experiments for training SAEs, capturing SAE activations and generating completions with Gemma 2 2B to be able to be run in about 30 A6000 hours. The LLM judging experiments take less than USD $20 through OpenRouter with Gemini 2.5 Flash Preview [22].

[9] `https://github.com/EleutherAI/sparsify`

[10] `https://wandb.ai`

D Broader Impact

Our work focuses on interpreting language models, an important component of building safer and more reliable systems. SAEs in particular are a popular technique for understanding language models, and through investigating dense latents, we can both better inform SAE design, and better understand language model internals.

We do not foresee any negative impacts of our work.

E Additional Taxonomy Results

E.1 Classification of dense latents

In our taxonomy, we identify dense latents using automated tests. We do not expect these tests to be perfect for a variety of reasons—for instance, dense latents not lining up perfectly with the "true" feature basis due to learning a linear combination basis, and the fundamental difficulty of designing true, causal tests. However, for the purposes of illustration, we choose reasonable cutoffs for each test to create Figure 2, listed below.

- Position latents: Spearman correlation of $|\rho| > 0.4$ for the relevant text boundary.
- Context-binding latents: Fraction of successful flips > 0.75.
- Nullspace latents: > 0.2 of encoder weight in bottom 10 $\mathbf{W}_\mathbf{U}$ singular vector subspace.
- Alphabet latents: Top 100 or bottom 100 logit contributions contain at least 90% of tokens starting with same character.
- Meaningful-word latents: AUC of using "is meaningful word" to predict "feature fires" > 0.75.
- PC-aligned latents: cosine similarity with top PC > 0.75.

A few dense latents fall in 2 categories based on our automated tests to find them, with the most common clashes being between sentence- and paragraph- tracking (see Appendix E.2), and between several categories and meaningful-word latent. For the purposes of illustration, we break ties according to the priority (from highest to lowest): {context-tracking, sentence-tracking, alphabet, nullspace, context-binding, paragraph-tracking, meaning, PCA} based on our confidence in our automated tests.

E.2 Position latents

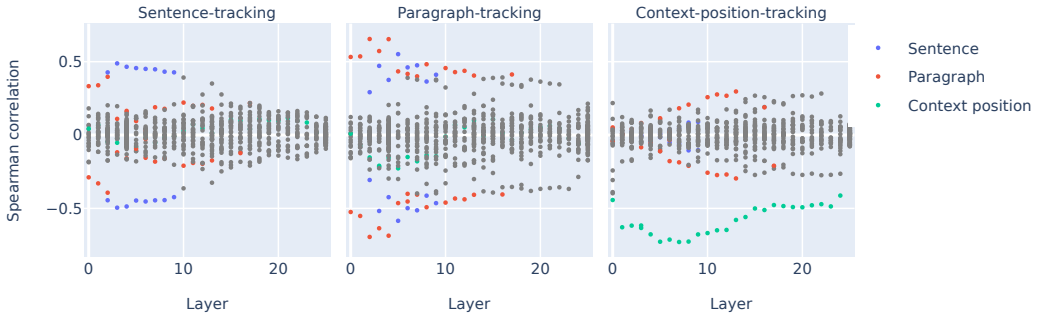

Figure 13: **Position Latents**. We identify position latents by computing their Spearman correlation ρ with relevant text boundaries. We classify a latent as belonging to a certain category when $|\rho| > 0.4$.

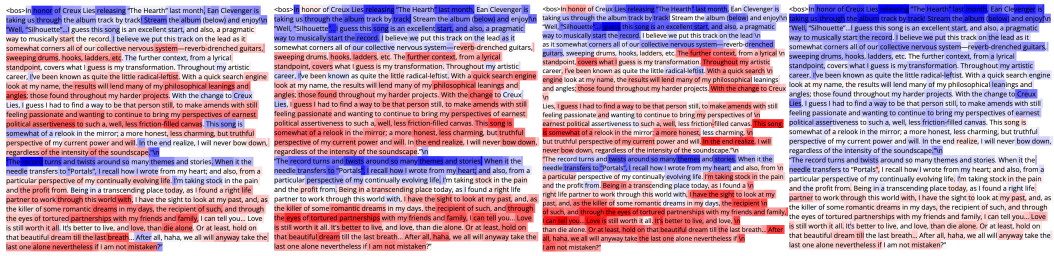

Figure 14: Examples of position latents in layer 5. Deep blue represents positive projection along decoder vector, and deep red represents negative. (1) L5:4341 is a sentence-tracking latent, that lights up consistently on beginnings of sentences. It has strong activations for topic sentences too. (2) L5:8680 is a paragraph-tracking latent, that lights up on beginnings of paragraphs. (3) L5:8680 is agnostic to artificially adding formatting newlines, showing it is encoding true paragraph position. (4) L5:697 is a context-position-tracking latent.

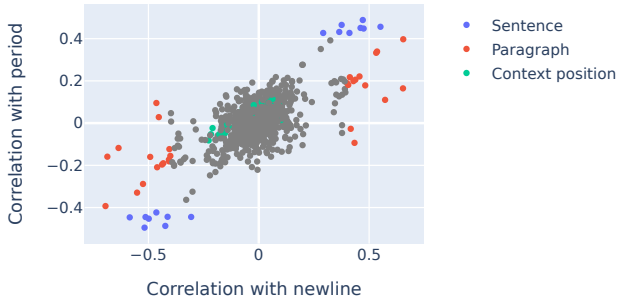

Figure 15: Spearman correlation for period against Spearman correlation for newline.

The observation in Figure 15 that period-tracking and newline-tracking latents are hard to distinguish also relates to our discussion in §6 that because the sparsity incentive is low for these dense latents, they may not be perfectly aligned to "true" model dense features, and may instead be a linear combination of two related features.

E.3 Additional examples of context-binding latents

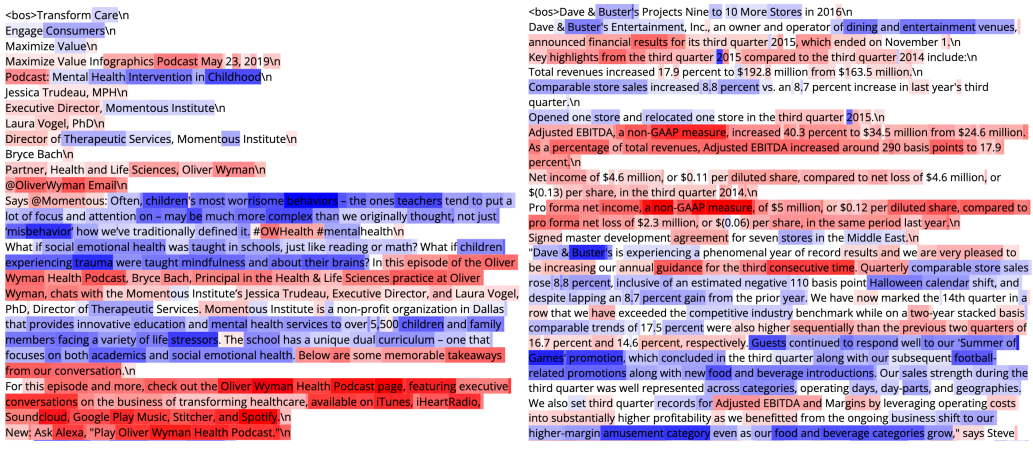

Figure 16: L13: 15275 (blue) and 11449 (red), which has 81.5% correct flips. In these two examples, 15275 fires on children's mental health (left) and Dave & Buster's promotions (right), while 11449 fires on mentions of the podcast (left) and financial measures (right).

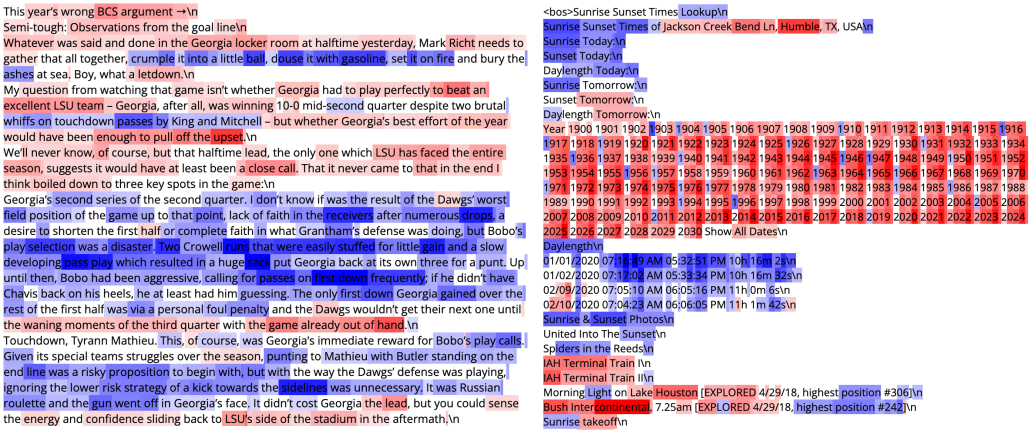

Figure 17: L12: 14906 (blue) and 14599 (red), which has 76.5% correct flips. In these two examples, 14906 fires on descriptions of the game (left), and text or numbers related to sunrise (right), while 14599 fires on the teams and winning/losing (left), and years or locations (right).

E.4 Steering context-binding latents

Our methodology for steering is as follows:

1. Prompt Gemma-2-2B with input text from the RedPajama dataset, ending at a natural point (after a newline token), with at least 400 tokens.

2. Capture the activating phrases of F1 and F2 that are at least 5 consecutive tokens long.

3. Allow Gemma-2-2B to generate a completion without steering, and prompt an LLM (Gemini 2.5 Flash Preview) to judge whether the completion is more like F1 activating examples, F2 activating examples, or unclear.

4. Repeat the above, but steering on the last token during generation, in the direction of F1 and F2. Since F1 and F2 are antipodal pairs, we first ablate the subspace spanned by F1 and F2, before adding the steering vector, that is fixed at 2x the historical activation of that feature in that context.

E.5 Interpreting context-binding latents

We attempt to interpret mid-layer latents that exhibit coherent chunk-level activations, by asking an LLM judge to generate natural language explanations based on activating examples. When examples are drawn from the same context, the explanations are often detailed and specific. However, when examples are drawn from different contexts, the explanations become vague or generic (Table 3).

This drop in specificity across contexts is somewhat expected, since explanations for any SAE feature may overfit the context. It is difficult to rule out the possibility that these dense latents represent an uninterpretable abstract feature the model learns. However, the causal steering experiment seems to cause the relevant specific concepts to be brought up during generation, supporting the "binding" hypothesis.

In-context explanations	Out-of-context explanations
References to violence, death, or controversial/negative events, either real or fictional	Discussion of a specific, named concept or entity within a broader category (e.g., a specific martial art, a specific plant, a specific architectural movement, a specific medical condition, a specific search tool)
Mentions of genres, particularly speculative fiction genres (sci-fi, fantasy, horror)	Lists or enumerations of related items or concepts
References to the Beautiful Creatures book series	Identifying or defining a role, title, or specific part of something
References to past events or states of being, often with a negative or challenging connotation	Describing a state of being in control, leading, or advancing
Accessing or observing information about others (competitors, other users, etc.)	References to specific, named entities or concepts within a larger domain (e.g., a specific TV show, a specific phrase, a specific team name)
References to famous or noteworthy artworks and their associated information (artist, price, era, nickname)	Referring to a specific concept or phenomenon by its name or a descriptive phrase
References to online platforms or marketplaces where user-generated content, fan-related items, or resale goods are exchanged	References to negative or undesirable actions/events/concepts
Trauma and its impact on mental health, particularly in children	Describing a specific curriculum, program, or set of agreements/targets
Mentions of the band Greta Van Fleet or their music	Referring to a specific action or event that is happening or has happened
References to totalitarian or authoritarian political ideologies and systems	References to specific, identifiable entities or concepts (products, organizations, actions, objects, people)

Table 3: LLM (Gemini 2.5 Flash Preview) generated explanations of suspected context-binding latent L12:7541, given 5 randomly drawn same-context examples vs 5 randomly drawn cross-context examples. 10 example explanations are shown.

E.6 Meaningful-Word Latents

In addition to the high-level categories shown in the main body, we found the AUC for several other categories, some of which are shown here.

Figure 18: From left to right, we show the AUCs of predicting latent firing using function words (any of {'article', 'prepos', 'conjunction', 'det', 'modal', 'be', 'do', 'have', 'what'}), articles, prepositions and conjunctions. These do not do as well as the "meaningful-word" or "noun/propernoun" groupings.

Category	Tags
punc	. ( ) * - , : “ ” '
quantifier	ABL, ABN, ABX, AP, AP$
article	AT
be	BE, BED, BEDZ, BEG, BEM, BEN, BER, BEZ
conjunction	CC, CS
num	CD, OD
do	DO, DOD, DOZ
det	DT, DTI, DTS, DTX, DT$
have	HV, HVD, HVG, HVN, HVZ
prepos	IN, TO
adj	JJ, JJR, JJS, JJT
modal	MD
noun	NN, NN$, NNS, NNS$, NR, NRS, NR$, UH
propernoun	NP, NP$, NPS, NPS$
pronoun	PN, PN$, PP$, PP$$, PPL, PPLS, PPO, PPS, PPSS
qual	QL, QLP
adv	RB, RB$, RBR, RBT, RN, RP
verb	VB, VBD, VBG, VBN, VBZ
what	WDT, WP$, WPO, WPS, WQL, WRB, EX
unknown	NIL

Table 4: Mapping from high-level category to Penn Treebank tags. A trailing $ marks possessive forms.

E.7 PCA latents

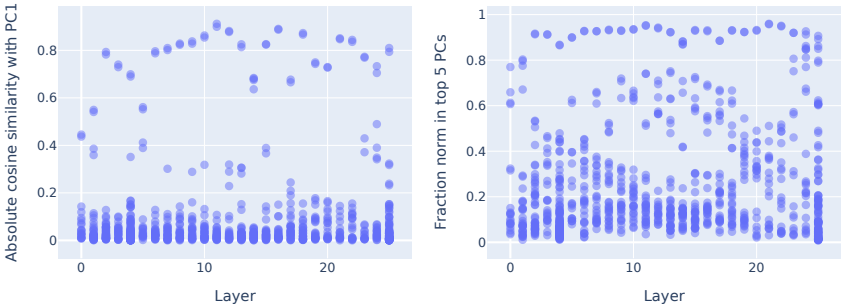

Figure 19: (Left) Cosine similarity of dense latents with top principal component. (Right) Fraction norm of dense latents in top 5 principal components.

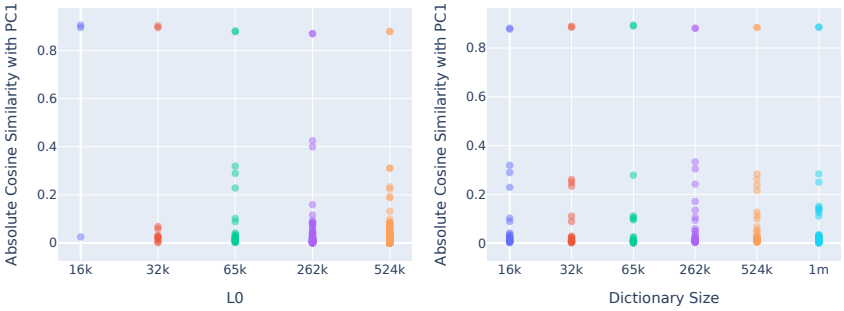

Figure 20: Cosine similarity of dense latents in layer 12 with the top principal component, across different L0s and SAE dictionary sizes.