INCORPORATING HIERARCHICAL SEMANTICS IN SPARSE AUTOENCODER ARCHITECTURES

Anonymous authors
Paper under double-blind review

ABSTRACT

Sparse dictionary learning (and, in particular, sparse autoencoders) attempts to learn a set of human-understandable concepts that can explain variation on an abstract space. A basic limitation of this approach is that it neither exploits nor represents the semantic relationships between the learned concepts. In this paper, we introduce a modified SAE architecture that explicitly models a semantic hierarchy of concepts. Application of this architecture to the internal representations of large language models shows both that semantic hierarchy can be learned, and that doing so improves both reconstruction and interpretability. Additionally, the architecture leads to significant improvements in computational efficiency.

1 INTRODUCTION

Dictionary learning—and, in particular, sparse autoencoders (SAEs)—have attracted significant attention as a tool for interpreting representations in large language models (Bricken et al., 2023; Cunningham et al., 2023; Gao et al., 2024; Lieberum et al., 2024; Lindsey et al., 2025b). The aim of these methods is to jointly learn some set of human-understandable concepts that can explain the model's behavior, and a map from the model's internal representations to these concepts. Empirically, at least some of the features learned by SAEs do seem clearly semantically interpretable—for example, "Golden Gate Claude" used an SAE feature to coherently steer Claude (Templeton et al., 2024). However, despite considerable effort, these models have some strong limitations. In particular, the learned features generally do not suffice to accurately reconstruct the input—limiting the value for interpretability—and, as the model size increases, the features that are learned often seem semantically unnatural. For example, Chanin et al. (2024) find that increasing the model size leads to a phenomenon they call "feature splitting"—where a single high-level concept is represented by multiple low-level features. The underlying tension here is that pushing for accurate reconstruction leads us to increase the number of features, but increasing the number of features can lead to very specialized features that are not generally useful.

The goal of this paper is to improve this reconstruction-interpretability frontier by exploiting the hierarchical structure of semantics. The motivating observation is that concepts that are meaningful to humans are often organized in a hierarchical fashion—for example, corgi, greyhound, and shitzu are all particular instances of the general concept of dog. Standard dictionary learning will not make use of this hierarchical structure at all (instead, representing each feature individually). Intuitively, we would like to modify the data structure such that this hierarchy is explicitly represented. The hope is that this will allow us to capture the reconstruction power of large dictionaries while maintaining interpretability by appealing to the human-understandable structure of the hierarchy.

The main contribution of this paper is a new architecture for SAEs that explicitly, and highly efficiently, models hierarchical relationships between concepts. Concretely:

1. Park et al. (2024a) give foundational results on how hierarchical structure is represented in language models. We show how to translate this theory into a mixture-of-experts type architecture (see Figure 2) that captures hierarchical structure (see Figure 1).
2. We then show that this architecture strongly improves the reconstruction performance of SAEs, while maintaining or improving interpretability.

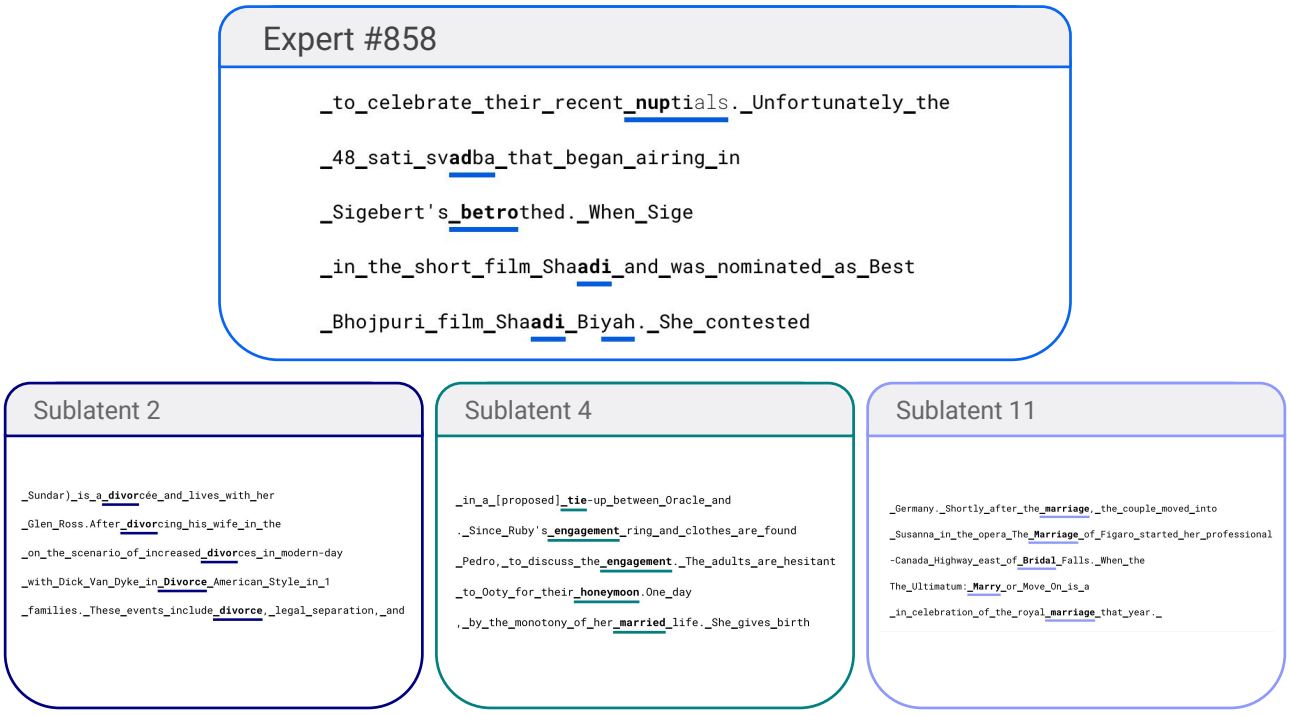


Figure 1: A Hierarchical Sparse Autoencoder architecture learns human interpretable hierarchy. Each box shows 5 of the strongest activating contexts for the feature, all tokens underlined activate the feature while the bold/text weight shows the relative strength of activation. For example, a "marriage" high-level feature and "divorce", "engagement", and "marriage" sublatents. Note that "48 sati svadba" is a Serbian wedding reality TV show and "Shaadi" is the Hindi word for marriage.

As an additional benefit, the new architecture is highly computationally efficient. In principle, this can allow scaling SAEs to much larger effective dictionary sizes, allowing fine-grained representations of a vast number of concepts.

2 Background and Related Work

Sparse Autoencoders SAEs are a particular approach to sparse dictionary learning, a general class of unsupervised learning algorithms that aim to find a dictionary of human-interpretable features (or atoms) that can be used to reconstruct the input data. The hope is that by enforcing sparsity we can recover some set of underlying latent factors. There are a large number of methods based on this idea.

We'll focus on top-k SAE approach of Gao et al. (2024), which has the advantages of being simple and scalable. The idea is to map each input vector x to a latent representation that is sparsified by a TopK operation that preserves the k largest values and sets all others to zero. Then, the vector is reconstructed by decoding this sparse latent representation. In total, the SAE operation is given by:

$$\mathrm{SAE}_k(\mathbf{x}) = \mathbf{D}\mathrm{TopK}_k\left(\mathrm{LeakyReLU}_\alpha\left(\mathbf{E}\left(\mathbf{x}-\mathbf{b}\right)\right)\right) \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^d$ is the input, $\mathbf{E}$ is the encoder matrix, $\mathbf{D}$ is the decoder matrix, $\mathbf{b}$ is a bias vector (with the rows, columns, and dimension of $\mathbf{E}$, $\mathbf{D}$, and $\mathbf{b}$ respectively being equal to the number of features) k is a hyperparameter, and $\alpha = \frac{1}{\sqrt{d}}$ is an adaptive threshold for the LeakyReLU activation (where LeakyReLU has some small negative slope below the threshold). The vectors in $\mathbf{D}$ are referred to as "features" or "latent vectors." The parameters are learned by minimizing the reconstruction loss:

$$\mathcal{L}_{\text{recon}} = \|\mathbf{x} - \mathrm{SAE}_k(\mathbf{x})\|_2^2 \tag{2}$$

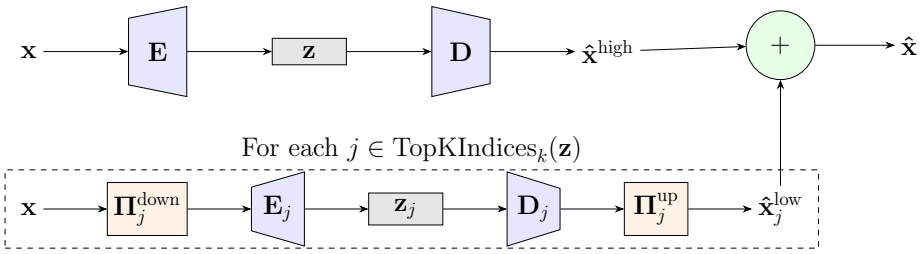


Figure 2: The structure of the Hierarchical Sparse Autoencoder architecture. The design combines a top-level encoder-decoder (upper path) that captures general concepts with expert-specific autoencoders (lower paths) that model refined features. For a given input, only a sparse subset of experts is activated, enhancing computational efficiency while maintaining expressive power.

Related Work The use of sparse autoencoders in LLM interpretability has been a topic of considerable interest (Marks et al., 2025), (Lindsey et al., 2025a), (Engels et al., 2024), (Kissane et al., 2024).

The most closely related is a line of work focused on ameliorating feature splitting. For instance, Matryoshka SAEs (Bussmann et al., 2025) and EWG-SAEs (Li & Ren, 2025) both operate by defining groupings of dictionary atoms and then modifying the training objective of the SAE to encourage some of the groupings to contain coarser features, and other groupings to contain finer features. These approaches are also motivated by the hierarchical nature of semantics. In contrast to the approach we take here, they do not modify the flat set structure of the architecture. Modifying the architecture has the advantages that it both makes the hierarchical structure explicit—improving interpretability—and also allows us to leverage the hierarchical structure to greatly improve computational efficiency, as we will see.

We also highlight Switch SAEs (Mudide et al., 2024), which introduce a mixture-of-experts type routing mechanism shares some structural similarities to our approach. However, this approach is entirely motivated by pushing the reconstruction quality vs sparsity frontier for a given compute budget. Each expert has no high vs low-level distinction and isn't designed to contain a set of features that should be interpreted together. Indeed, they find that their decoder features do not cluster in any particular way.

The ideas here connect to the broader field of causal representation learning (CRL), which aims to learn the causal factors underlying a data-generating process (Locatello et al., 2019; 2020; Schölkopf et al., 2021; Ahuja et al., 2022a; Brehmer et al., 2022; Lippe et al., 2023; Moran & Aragam, 2025). Recent work has extended CRL to foundation model analysis, seeking identifiable and disentangled representations of concepts (Rajendran et al., 2024; Joshi et al., 2025). In this spirit, we also aim to learn disentangled concept representations.

3 Hierarchical Sparse Autoencoder Architecture

We now turn to the development of an architecture that bakes in the hierarchical structure of concepts.

3.1 Architecture

Hierarchical Geometry Our main inspiration follows Park et al. (2024a), who find that the representations of categorical concepts in language models have a specific geometric structure. In particular, they show that every categorical concept has *two* representations: a parent feature indicating whether the concept is active, and a low-rank subspace of the representation space containing a polytope where each point is a child features corresponding to a different possible value of the concept. For example, the concept "dog" is represented by a parent feature indicating whether the concept is active ('is dog' vs 'not dog'), and a low-dimensional subspace containing a polytope where each point is a child feature corresponding to a different breed of dog.

The idea is to create an architecture that matches this dual representation structure; see Figure 2. To achieve this, we propose an architecture with three parts:

1. A top-level SAE that aims to capture the binary feature indicating whether a high-level concept is active. This is a standard SAE with a relatively small number of features.
2. For each atom in the high-level SAE, we have an associated down projector onto a low-dimensional subspace (and an up-projector mapping back to the original space). This corresponds to the low-dimensional subspace associated to the categorical concept. Since each high-level concept spans a low-dimensional subspace, the only requirement for the 'projection' is staying within the column space of the concept. Thus, we do not impose idempotence or symmetry and simply use 2 trainable matrices.
3. For each atom in the high-level SAE, we have a low-level SAE that operates on the low-dimensional subspace associated with atom. The elements of each such SAE correspond to the different possible values of the categorical concept (e.g., the different breeds of dog).

That is, the architecture is a high-level SAE where each atom also has a low-level SAE attached to it.

Now, a critical observation here is that an atom in the low-level SAE can only be active if the high-level atom is also active. That is, for the residual stream vector x to encode the concept of "corgi" it *must* also encode the concept of "dog." To respect this constraint, we structure the model as a mixture-of-experts type architecture, where a low-level SAE is only called if the associated high-level feature is active. This both respects the hierarchical structure of the concepts and also allows for an enormous computational efficiency gain (see below).

In fact, the Park et al. (2024a) results are more precise than the informal presentation above. In particular, they show that a low level concept ("corgi") is naturally represented as the *sum* of a vector for the high-level concept ("dog") and a vector representing the low-level concept in the context of the high level space (i.e., "corgi" = "dog" + "corgi | dog"). It is these contextual representations that live in a low-dimensional space. This overall structure—a low level concept is represented as the sum of a high-level concept plus the low-level concept in the context of the high-level one—is exactly the structure implemented by the H-SAE.

Forward Pass We can now make the architecture precise:

$$\text{H-SAE}(\mathbf{x}) = \sum_{j \in \text{TopKIndices}_k} \Big(\underbrace{z_j \mathbf{d_j}}_{\hat{\mathbf{x}}^{\text{high}}} + \underbrace{\mathbf{\Pi_j^{up}} \text{SAE}_1^j(\mathbf{\Pi_j^{down}} \mathbf{x})}_{\hat{\mathbf{x}}_j^{\text{low}}} \Big) \tag{3}$$

where TopKIndices_k are the indices of the top k features, $\mathbf{d_j}$ is the j-th high-level feature, $z_j = (\text{LeakyReLU}(\mathbf{Ex}))_j$ is the corresponding code, SAE_1^j is the expert-specific autoencoder for high-level feature j, and $\mathbf{\Pi_{up}^j}$ and $\mathbf{\Pi_{down}^j}$ are projection matrices (of dimension s) that map the input to the expert subspace and back to the original space, respectively. Note that the low-level SAE uses TopK_1 over its a features, retaining only a single low-level feature for each expert. This corresponds to the idea that the representation of a subordinate concept should be at a particular vertex in the polytope corresponding to the categorical concept.

We depict this architecture visually in Figure 2 and algorithmically in Algorithm 1.

Computational Efficiency Beyond explicitly enforcing the target semantic structure, activating the low-level SAE only when the corresponding high-level feature is active provides a significant computational advantage. The computational cost of a forward pass can be expressed as follows:

$$\begin{aligned} \text{Cost}_{\text{forward}} = & \underbrace{O(jd)}_{\text{High-level encoding}} + \underbrace{O(ksd)}_{\text{Subspace projection}} + \underbrace{O(kas)}_{\text{Low-level encoding}} \\ & + \underbrace{O(kas)}_{\text{Low-level decoding}} + \underbrace{O(ksd)}_{\text{Upward projection}} + \underbrace{O(kd)}_{\text{High-level decoding}} \end{aligned}$$

where j is the number of high-level features (experts), d is the dimensionality of the input activation vector, s is the subspace dimension for each expert, a is the number of low-level features per expert, and k is the sparsity level (number of activated experts). In the typical case where $k \ll j$ (sparsity) and $s \ll d$ (low dimensional subspace) the cost is dominated by the top-level SAE. That is, equipping

Algorithm 1 Hierarchical SAE Forward Pass and Loss Computation

function ForwardPass($\mathbf{x}$)
  $\mathbf{z} \leftarrow \text{TopK}_k(\text{LeakyReLU}_\alpha(\mathbf{E}\mathbf{x}))$ ▷ High-level encoding
  $\mathcal{K} \leftarrow$ indices of nonzero elements in $\mathbf{z}$
  $\hat{\mathbf{x}}^{\text{high}} \leftarrow \mathbf{D}\mathbf{z}$ ▷ High-level reconstruction
  $\hat{\mathbf{x}}^{\text{low}} \leftarrow \mathbf{0}$ ▷ Initialize low-level reconstruction
  for $j \in \mathcal{K}$ **do** ▷ Only process activated experts
    $\mathbf{x}_j^{\text{sub}} \leftarrow \mathbf{\Pi}_j^{\text{down}}\mathbf{x}$ ▷ Project to expert subspace
    $\mathbf{z}_j \leftarrow \text{LeakyReLU}_\alpha(\mathbf{E}_j\mathbf{x}_j^{\text{sub}})$ ▷ Low-level encoding
    $\hat{\mathbf{x}}_j^{\text{sub}} \leftarrow \mathbf{D}_j\mathbf{z}_j$ ▷ Reconstruct in subspace
    $\hat{\mathbf{x}}^{\text{low}} \leftarrow \hat{\mathbf{x}}^{\text{low}} + \mathbf{\Pi}_j^{\text{up}}\hat{\mathbf{x}}_j^{\text{sub}}$ ▷ Project back and accumulate
  end for
  $\hat{\mathbf{x}} \leftarrow \hat{\mathbf{x}}^{\text{high}} + \hat{\mathbf{x}}^{\text{low}}$ ▷ Combined reconstruction
  return $\hat{\mathbf{x}}, \mathbf{z}, \{\mathbf{z}_j\}_{j\in\mathcal{K}}$
end function

function ComputeLoss($\mathbf{x}, \hat{\mathbf{x}}, \mathbf{z}, \{\mathbf{z}_j\}_{j\in\mathcal{K}}$)
  $\mathcal{L}_{\text{recon}} \leftarrow \|\mathbf{x} - \hat{\mathbf{x}}\|_2^2$
  $\mathcal{L}_{\text{top\_recon}} \leftarrow \|\mathbf{x} - \mathbf{D}\mathbf{z}\|_2^2$
  $\mathcal{L}_{\text{recon}} \leftarrow \mathcal{L}_{\text{recon}} + \beta\mathcal{L}_{\text{top\_recon}}$ ▷ Encourage meaningful top-level features
  $\mathcal{L}_{\text{sparse}} \leftarrow \|\mathbf{z}\|_1 + \sum_{j\in\mathcal{K}} \|\mathbf{z}_j\|_1$
  $\mathcal{L}_{\text{ortho}} \leftarrow \frac{\|\mathbf{DE} - \text{diag}(\mathbf{DE})\|_F}{n^2-n}$
  $\mathcal{L} \leftarrow \mathcal{L}_{\text{recon}} + \lambda_1\mathcal{L}_{\text{ortho}} + \lambda_2\mathcal{L}_{\text{sparse}}$
  return $\mathcal{L}$
end function

the top-level SAE with hierarchical structure adds negligible computational overhead. Because the hierarchical SAE is much more expressive, this significantly improves SAE scalability. We also note that because the memory cost of a batch gradient step scales with the number of activated parameters, not the total number of parameters, this efficiency also applies to (per-step) training. Indeed, in practice we find that the additional cost imposed by the hierarchical structure is very small.

3.2 Training Objective

We also make some minor modifications to the training objective, using the following loss function:

$$\mathcal{L} = \mathcal{L}_{\text{recon}} + \lambda_1\mathcal{L}_{\text{ortho}} + \lambda_2\mathcal{L}_{\text{sparse}} \tag{4}$$

Reconstruction Loss Following standard practice, we measure reconstruction loss with Euclidean distance. Additionally, to encourage the top-level SAE features to be meaningful in their own right (rather than just as routers) we also penalize reconstruction error from the top-level SAE only. The reconstruction loss is then:

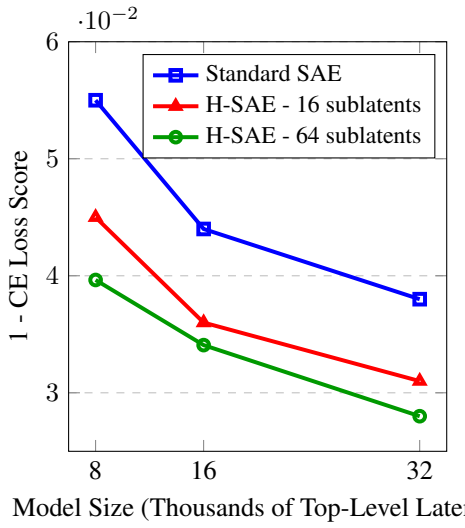
$$\mathcal{L}_{\text{recon}} = \|\mathbf{x} - \text{H-SAE}(\mathbf{x})\|_2^2 + \beta\|\mathbf{x} - \hat{\mathbf{x}}^{\text{high}}\|_2^2, \tag{5}$$

where $\beta = 0.1$ was chosen because the top-level SAE alone has less capacity than the full H-SAE.

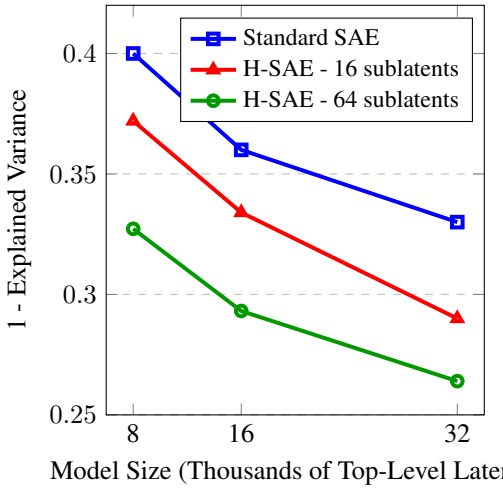
Auxiliary Losses Park et al. (2024b) show that the representations of "causally separable" (individually manipulable) features are bi-orthogonal in a certain sense.[1] This suggests that we could discourage semantic redundancy in the learned features (e.g., having multiple features for "dog") by imposing a suitable orthogonality constraint on the learned features. To operationalize this, we introduce a bi-orthogonality penalty on the top-level features:

$$\mathcal{L}_{\text{ortho}} = \frac{\|\mathbf{DE} - \text{diag}(\mathbf{DE})\|_F}{n^2 - n}, \tag{6}$$

[1] Namely, the dot product between primal and dual space representations is zero.



(a) The H-SAE has better reconstruction performance than a standard SAE as measured 1 - CE Loss score across different model sizes and with both 16 and 64 sublatents per expert. Lower values indicate better downstream model performance, using the SAEBench CE loss score.



(b) The H-SAE has better reconstruction performance than a standard SAE as measured 1 - Explained Variance across different model sizes and with both 16 and 64 sublatents per expert. Lower values indicate better capture of the data's inherent structure.

Figure 3: H-SAEs have better reconstruction performance as measured by explained variance and the downstream CE loss of a Gemma 2-2B using SAE-reconstructed activation vectors.

where $\text{diag}(\mathbf{DE})$ is the diagonal matrix formed by the diagonal elements of the top-level decoder multiplied by the top-level encoder, and $\|\cdot\|_F$ denotes the Frobenius norm. This pushes the encoder representation for feature i and decoder representation for feature j to be orthogonal if $i \neq j$. Mechanically, this means that if we ran the encoder on the autoencoder's own output, whether feature z_i was active in the first encoding would not effect feature z_j in the second encoding.

The motivation here is to encourage compositionality in the learned features. However, it is not clear how to empirically measure compositionality, and so we are unsure whether this term actually achieves this goal. Nevertheless, we do observe empirically that adding this term does an excellent job of mitigating the "dead atom" phenomena where many features are never used in reconstructions. We use this instead of the auxiliary dead latent loss proposed by Gao et al. (2024), but we do not believe this is a crucial component. See Section B for ablations.

We also include a small ℓ_1 sparsity penalty on the latent values on both the top and low-level features outside the top k to further encourage specialization; this is the $\mathcal{L}_{\text{sparse}}$ term. Again, it's unclear how to measure the target compositionality effect, and we do not believe this is a key component. See Section B for ablations.

4 Experiments

The main motivation for this work is that by incorporating the hierarchical structure of semantics we can improve the reconstruction-interpretability frontier. Accordingly, we want to answer three questions:

1. Does the H-SAE improve reconstruction?
2. Does the H-SAE maintain, or improve, interpretability?
3. Does the model in fact learn hierarchical semantics?

We will see that the answer to all three questions is yes.

Experimental Setup Our training data consists of 1 billion residual stream vectors extracted from layer 20 of Gemma 2-2B . We collect this data by running Gemma on a large corpus of Wikipedia

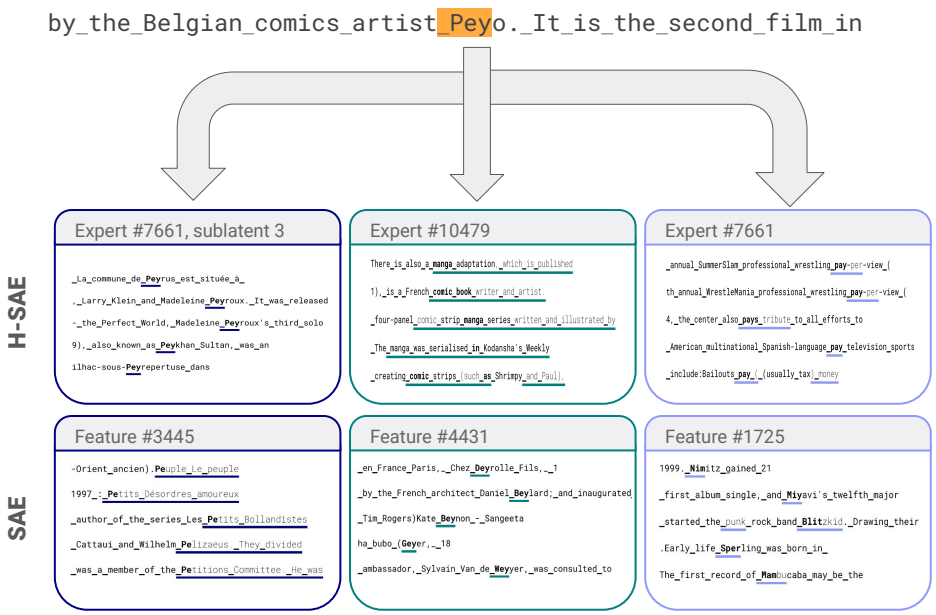


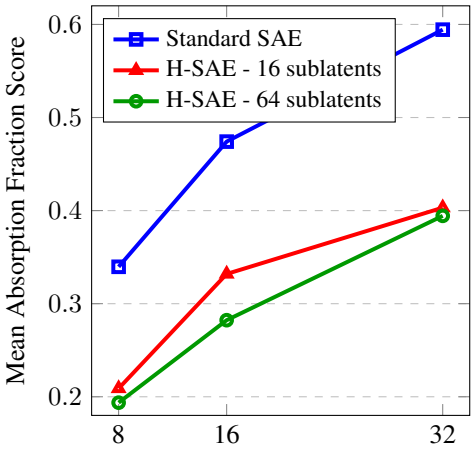
Figure 4: A comparison decomposition of the same context/embedding in the H-SAE architecture and standard SAE. The H-SAE represents a token about the creator of the Smurfs cartoon using a high level "Pay" homophone feature, and a low-level "Pey" token feature. This feature is then composed with a comic books feature. Wheras, the SAE has features for "words that end with 'ey'" and "Pe" words, but none of the top-32 contain a high-level feature for the 'comic book' context of this sentence, rather the SAE uses a 'part of an artists' name feature to reconstruct the embedding.

articles, spanning multiple languages and a wide range of topics. For each article, we extract the residual stream vectors corresponding to the first 256 tokens. Following standard practice, we normalize the vectors to unit norm (Lieberum et al., 2024). However, we do not subtract any mean vector nor include a bias term as Gao et al. (2024) do, as we found this decreased training stability.

As a baseline, we use the TopK SAE architecture (Gao et al., 2024). The H-SAE is trained using the objective function described in Equation (4). All models are trained on the same data for 4 epochs. The baseline top-k autoencoders empirically perform equivalently on reconstruction loss to the Gemma Scope JumpReLU autoencoders (Lieberum et al., 2024). See Section A for more details on training.

Reconstruction Figure 3 shows the reconstruction performance of the H-SAE and the baseline at a variety of dictionary sizes. We measure both 1 - explained variance (i.e. normalized ℓ^2 reconstruction loss) and the LLM CrossEntropy loss induced by replacing the original activations with the reconstructions (normalized by SAEBench between 0 and 1). As expected, adding hierarchical capacity to the model improves reconstruction performance. Indeed, this effect is dramatic. The H-SAE with 64 sublatents per expert is on par with the standard SAE with 32 top-level features, despite having $1/4$ the compute cost.

Matryoshka SAE Matryoshka SAEs (Bussmann et al., 2025) are the most similar SAE architecture conceptually, hoping to reduce undesirable feature absorption and increase hierarchical interpretability. To compare these models, we start by training them on the token unembeddings, as in other ablations. The features are fairly similar, with the H-SAE often finding sparser representations, finding high-level features the M-SAE does not, and fewer uninterpretable features. These factors also combine with the reconstruction loss and computational advantage the H-SAE has over both an M-SAE and standard SAE. Additionally, we train both an H-SAE and M-SAE on the synthetic benchmark from the M-SAE paper, and find the H-SAE outperforms the M-SAE. Full results and experimental details are in Section D.



(a) The standard SAE shows higher absorption, indicating greater feature splitting, while our H-SAE architecture maintains more coherent representations as measured by the mean fraction of first-letter classifications that show signs of absorption.

Language Pair	H-SAE	SAE
English vs French	**8.448**	9.772
English vs Spanish	**8.190**	9.598
English vs German	**9.372**	10.938
French vs Spanish	**5.748**	7.170
French vs German	**7.306**	9.038
Spanish vs German	**7.476**	9.270

(b) The H-SAE architecture has less divergence in features for the same sentence in different languages. Higher values indicate more redundant features, as measured by the mean set difference between the same token in different languages, demonstrating our H-SAE architecture's superior ability to learn highly composable features.

Figure 5: H-SAEs exhibit less undesirable feature absorption and fewer redundant features across languages

4.1 Interpretability

By itself, an improvement in reconstruction performance may not be meaningful. The reason is that there are many possible modifications that improve reconstruction performance by sacrificing interpretability. This concern is somewhat mitigated by the fact that the extra capacity of the H-SAE is highly constrained; low-level features can only be activated when their corresponding high-level feature is active. Nevertheless, we would like to directly evaluate the interpretability of the H-SAE against a standard SAE.

Figure 4 shows a comparison of the baseline and H-SAE decomposition of the same context. We observe that the H-SAE has features that are of the same quality, if not better. This behavior is typical. See Section C for more examples of features and supplementary materials for an interactive notebook including visualizations of hundreds of features from each model.

To complement the visualizations, we perform some more systematic tests. Recent work has shown that general-purpose automated interpretability evaluations are misleading–particularly regarding more abstract features (Heap et al., 2025). Accordingly, we focus in particular on feature splitting and absorption. These are aspects that we would expect to see the greatest effect on through learning more general top-level features, and are relatively measurable. To test feature absorption, we run the first letter classification benchmark from SAEBench (Karvonen et al., 2025). This benchmark tests the improvement in probing performance for a first letter classification task as the number of features used increases above 1. This task should only require a single feature if there is no undesirable splitting or absorption. As expected, we find that the H-SAE architecture both improves performance on this task and decreases the rate of increase in absorption as width increases. See Figure 5a, where we report the fraction of the projection from SAE activations onto a first-letter classification probe that is not explained by a single feature.

We can also test for the presence of duplicate features or 'redundancy'. One particular example of this that we and others observe is equivalent syntactic features from different languages (e.g. a French period and an English period) (Lindsey et al., 2025b). So, we sample 1,000 English sequences from the training data, consisting of between 16 and 48 tokens. We then use Gemini-2.0-Flash to translate these sentences into French, Spanish, and German (the most common non-English languages in the training data). Then, we collect the top 8 most strongly activated features on the last token of the context for the standard and H-SAE. Ideally, the features activated by the same token in different

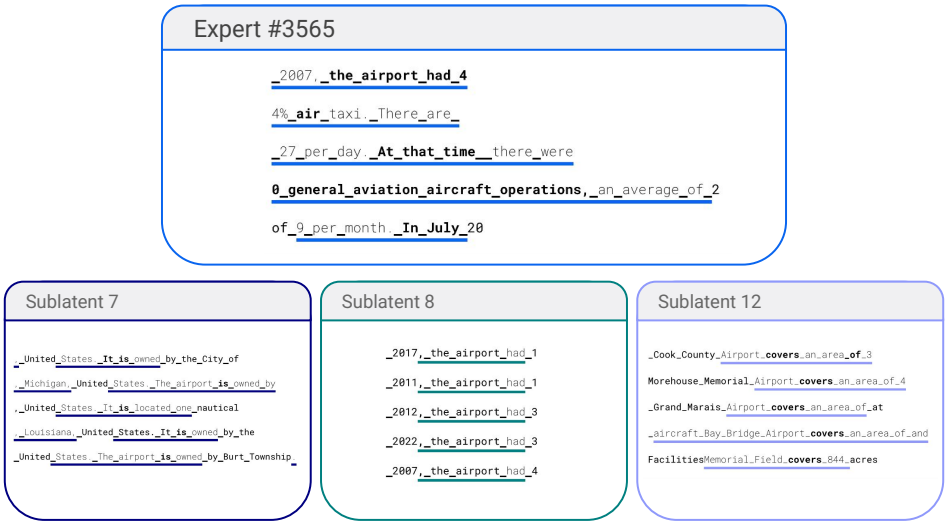


Figure 6: The H-SAE has a "airports" high-level feature and "US airport", "the token 'airport'", and "airport size" sublatents.

languages should be similar. We measure the size of the symmetric set difference between the top 8 features activated by the same token in different languages (a value between 0 and 16); see Figure 5b. As expected, the H-SAE uses more similar sets of features across all language pairs than the baseline SAEs.

Hierarchical Semantics Finally, we turn to the question of whether the H-SAE is indeed learning hierarchical semantics. To this end, we visualize examples where low-level features are strongly activated. Figures 1, 4 and 6 show examples from the 16k experts x 16 sublatents H-SAE. We observe that hierarchical semantics are clearly emerging. See the supplementary materials for an interactive notebook including hundreds of such visualizations.

5 Discussion

The main question in this work is whether the interpretability-reconstruction frontier can be improved by exploiting the hierarchical structure of semantics. We find that the answer is yes. Incorporating hierarchy dramatically improves reconstruction and moderately improves interpretability of the top-level features. Further, the two-level structure effectively communicates semantic relationships between concepts—e.g., we observed high-level latents capturing regional concepts like "Bay Area," with corresponding expert-specific sub-latents representing entities like "Stanford" that exist within that region. As a further benefit, the hierarchical structure allows for large increases in the computational efficiency relative to the effective number of atoms.

Limitations and Further Work In this work we primarily analyze the architectural changes in the context of a simple 'standard' SAE approach. Incorporating hierarchy leads to an improvement, but the results are still imperfect—e.g., we still find some level hard-to-interpret features, and we still far short of perfect reconstruction. It is unclear whether this reflects a fundamental limitation of dictionary learning, or whether there is some additional set of changes that would lead to dramatically improved performance. For example, our ablation studies on word unembeddings suggest that using a reconstruction objective other than Euclidean distance can massively improve interpretability; see Section B.3. Similarly, work in causal representation learning has shown simple sparsity objectives have fundamental limitations (Locatello et al., 2019), and developed a variety of more sophisticated objectives leading to much better performance (Locatello et al., 2020; Ahuja et al., 2022b; Lippe et al., 2022). It would be exciting to find ways to incorporate such insights into LLM interpretability.

REPRODUCABILITY

All the main experiments and results are reproducible using the code and data provided in the supplementary material. The full dataset used for training is too large to include directly as it is 10TB, but the full code used to create it is included. Model checkpoints are provided for all H-SAEs and standard SAEs trained. To aid with evaluation, supplementary material also includes hundreds of randomly selected features to demonstrate the consistent interpretability and quality of the learned features.

ACKNOWLEDGEMENTS

REFERENCES

Kartik Ahuja, Jason Hartford, and Yoshua Bengio. Weakly supervised representation learning with sparse perturbations, 2022a. URL https://arxiv.org/abs/2206.01101.

Kartik Ahuja, Jason S. Hartford, and Yoshua Bengio. Weakly Supervised Representation Learning with Sparse Perturbations. *Advances in Neural Information Processing Systems*, 35:15516–15528, December 2022b. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/63d3bae2c1f525745003f679e45bcf7b-Abstract-Conference.html.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. Jax: composable transformations of python+numpy programs. Computer software, 2018. URL http://github.com/jax-ml/jax.

Johann Brehmer, Pim de Haan, Phillip Lippe, and Taco Cohen. Weakly supervised causal representation learning, 2022. URL https://arxiv.org/abs/2203.16437.

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. https://transformer-circuits.pub/2023/monosemantic-features/index.html.

Bart Bussmann, Noa Nabeshima, Adam Karvonen, and Neel Nanda. Learning multi-level features with matryoshka sparse autoencoders, 2025. URL https://arxiv.org/abs/2503.17547.

David Chanin, Julian Wilken-Smith, Tomas Dulka, Harsh Bhatnagar, and Joseph Bloom. A is for absorption: Studying feature splitting and absorption in sparse autoencoders, 2024. URL https://doi.org/10.48550/arXiv.2409.14507.

Hoagy Cunningham, Aidan Ewart, Logan Riggs, Robert Huben, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models, 2023. URL https://arxiv.org/abs/2309.08600.

Joshua Engels, Isaac Liao, Eric J. Michaud, Wes Gurnee, and Max Tegmark. Not All Language Model Features Are Linear, May 2024. URL http://arxiv.org/abs/2405.14860. arXiv:2405.14860 [cs].

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.

Thomas Heap, Tim Lawson, Lucy Farnik, and Laurence Aitchison. Sparse Autoencoders Can Interpret Randomly Initialized Transformers, January 2025. URL http://arxiv.org/abs/2501.17727. arXiv:2501.17727 [cs].

David D. Johnson. Penzai + treescope: A toolkit for interpreting, visualizing, and editing models as data, 2024.

Shruti Joshi, Andrea Dittadi, Sébastien Lachapelle, and Dhanya Sridhar. Identifiable steering via sparse autoencoding of multi-concept shifts, 2025. URL https://arxiv.org/abs/2502.12179.

Adam Karvonen, Can Rager, Johnny Lin, Curt Tigges, Joseph Bloom, David Chanin, Yeu-Tong Lau, Eoin Farrell, Callum McDougall, Kola Ayonrinde, Matthew Wearden, Arthur Conmy, Samuel Marks, and Neel Nanda. Saebench: A comprehensive benchmark for sparse autoencoders in language model interpretability, 2025. URL https://doi.org/10.48550/arXiv.2503.09532.

Patrick Kidger and Cristian Garcia. Equinox: Neural networks in jax via callable pytrees and filtered transformations, 2021. URL https://doi.org/10.48550/arXiv.2111.00254.

Connor Kissane, Robert Krzyzanowski, Joseph Isaac Bloom, Arthur Conmy, and Neel Nanda. Interpreting Attention Layer Outputs with Sparse Autoencoders, June 2024. URL http://arxiv.org/abs/2406.17759. arXiv:2406.17759 [cs].

Ed Li and Junyu Ren. UNLOCKING HIERARCHICAL CONCEPT DISCOVERY IN LANGUAGE MODELS THROUGH GEOMETRIC REGULARIZATION. In *ICLR 2025 Workshop on Building Trust in Language Models and Applications*, 2025. URL https://openreview.net/forum?id=i31cKXiyim.

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

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

Jack Lindsey, Will Gurnee, Emmanuel Ameisen, Brian Chen, Adam Pearce, Nicholas L. Turner, Chris Citro, Daniel Abrahams, Shan Carter, Ben Hosmer, Jonathan Marcus, Michael Sklar, Adly Templeton, Trenton Bricken, Callum McDougall, Hoagy Cunningham, Tom Henighan, Adam Jermyn, Andy Jones, and Joshua Batson. On the biology of a large language model, 2025b. URL https://transformer-circuits.pub/2025/attribution-graphs/biology.html.

Phillip Lippe, Sara Magliacane, Sindy Löwe, Yuki M. Asano, Taco Cohen, and Stratis Gavves. CITRIS: Causal Identifiability from Temporal Intervened Sequences. In *Proceedings of the 39th International Conference on Machine Learning*, pp. 13557–13603. PMLR, June 2022. URL https://proceedings.mlr.press/v162/lippe22a.html. ISSN: 2640-3498.

Phillip Lippe, Sara Magliacane, Sindy Löwe, Yuki M. Asano, Taco Cohen, and Efstratios Gavves. Biscuit: Causal representation learning from binary interactions, 2023. URL https://arxiv.org/abs/2306.09643.

Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Rätsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations, 2019. URL https://arxiv.org/abs/1811.12359.

Francesco Locatello, Ben Poole, Gunnar Rätsch, Bernhard Schölkopf, Olivier Bachem, and Michael Tschannen. Weakly-supervised disentanglement without compromises, 2020. URL https://arxiv.org/abs/2002.02886.

Jeremy Maitin-Shepard and Laramie Leavitt. Tensorstore for high-performance, scalable array storage. Computer software, 2022. URL https://github.com/google/tensorstore.

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, 2025. URL https://arxiv.org/abs/2403.19647.

Gemma E. Moran and Bryon Aragam. Towards interpretable deep generative models via causal representation learning, 2025. URL https://arxiv.org/abs/2504.11609.

Anish Mudide, Joshua Engels, Eric J. Michaud, Max Tegmark, and Christian Schroeder de Witt. Efficient dictionary learning with switch sparse autoencoders, 2024. URL https://arxiv.org/abs/2410.08201.

Kiho Park, Yo Joong Choe, Yibo Jiang, and Victor Veitch. The geometry of categorical and hierarchical concepts in large language models. *arXiv preprint arXiv:2406.01506*, 2024a.

Kiho Park, Yo Joong Choe, and Victor Veitch. The linear representation hypothesis and the geometry of large language models. In *Forty-first International Conference on Machine Learning*, 2024b.

Goutham Rajendran, Simon Buchholz, Bryon Aragam, Bernhard Schölkopf, and Pradeep Ravikumar. Learning interpretable concepts: Unifying causal representation learning and foundation models, 2024. URL https://arxiv.org/abs/2402.09236.

Bernhard Schölkopf, Francesco Locatello, Stefan Bauer, Nan Rosemary Ke, Nal Kalchbrenner, Anirudh Goyal, and Yoshua Bengio. Towards causal representation learning, 2021. URL https://arxiv.org/abs/2102.11107.

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. URL https://transformer-circuits.pub/2024/scaling-monosemanticity/index.html.

A Training Details

A.1 Input Data Construction

- The 20231101 Wikipedia dump is used to construct the data. Approximately 500 million English tokens are selected by taking the first 256 tokens among articles on English and Simple English Wikipedia. The articles are selected by choosing the top 2 million longest articles (using length as a proxy for quality to filter out low quality stub articles) and then randomly among those articles until 500 million tokens have been selected.
- For non-English tokens, articles are chosen randomly from the largest 128 non-English Wikipedias (as measured by active users). The mix of languages is also proportional to the number of active users. This is used as a rough proxy for Wikipedia quality.
- Tokens are then packed into sequences to fill the Gemma 2-2B context window and activations are collected using Penzai and stored on disk with Tensorstore (Johnson, 2024; Maitin-Shepard & Leavitt, 2022).
- BOS and EOS tokens are stripped from the data and shuffled on disk prior to training. SAE training requires data access well in excess of 1GB/s to saturate a 8xA100 node and so the data must be shuffled ahead of time.

A.2 Model Implementation and Hyperparameters

- The hierarchical sparse autoencoder is implemented in JAX and Equinox (Bradbury et al., 2018; Kidger & Garcia, 2021).
- We train with a batch size of 32,512 and top-k of 32. When the number of sublatents per expert is 16, the subspace dimension is 4. Otherwise, it is 8.

- We set the orthogonality penalty and top-level reconstruction coefficients to 0.1. The l1 coefficient is 0.001. For standard SAEs that require an auxiliary loss to prevent dead latents we use a coefficient of $1/30$.
- We use the adam optimizer with global norm clipping of 0.75 and b1 of 0.9.
- The learning rate is $5 \cdot 10^{-4}$, initialized to 10^{-11} with a 1,000 steps linear warmup and cosine decay over the remainder of the training run.
- Additionally, the regularizers also warmup from 0 over the first 1,000 steps.

A.3 Training Dynamics

- We track an exponential moving average over 300 batches (i.e. roughly every million tokens) of the number of times latents activate. Both the auxillary loss and orthogonality loss push the number of latents that don't activate in 1 million tokens (i.e. dead) well under 1%. The dead atom auxillary loss coincidentally has exactly the same compute cost as the addition of hierarchy with the hyperparameters used for experiments. However, in terms of wall-time the H-SAE actually trains slightly faster due to the auxillary loss being a much more memory-intensive operation using a naive implementation. We do not attempt to use maximally efficient implementations so detailed compute analysis is not conducted.

B Ablations

B.1 Pre-Processing

We conduct ablations by evaluating our H-SAE architecture applied to the token unembedding matrix of the Gemma 2-2B model, rather than internal activations, as the unembedding matrix provides a small data set that is more suitable for ablation studies. Gemma 2-2B has 256,128 tokens in its vocabulary, however many of these are relatively rare and meaningless tokens. After discarding these, we are left with 186,032 tokens.

Previous work (Park et al., 2024b) has shown that the semantic structure of the unembedding matrix is fundamentally non-Euclidean, meaning that the standard Euclidean inner product does not align orthogonality with semantic independence. Following this line of work, we whiten the unembedding matrix by multiplying it by the inverse square root of the covariance matrix, which has been shown to align the inner product with the underlying geometry of the data. Empirically, we observe that this whitening step is necessary for the model to learn meaningful features.

B.2 Ablation Setup

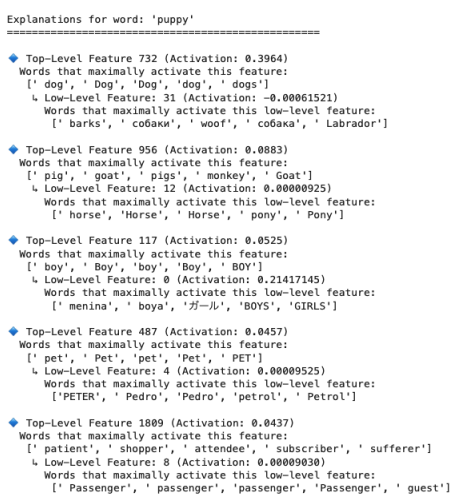
For the ablations, we apply our H-SAE architecture to this whitened matrix, using a top-level dictionary with 2048 latent vectors (experts) and 32 sublatents per expert, $k = 5$. We considered an ℓ_1 strength of 2.5×10^{-3} and an orthogonality penalty of 2.5×10^{-2}. As our focus was on ablations, we were not concerned with the absolute performance of the model, but rather with the relative performance of different configurations, so we did not tune hyperparameters beyond those being compared. We trained for $10,000$ steps with a batch size of 8192 and a cosine-decay learning rate schedule starting at 8192×10^{-15} and peak value of 8192×10^{-6}.

We are interested in whether the orthogonality loss and ℓ_1 regularizer are necessary for interpretability and hierarchy. Because this is fundamentally a qualitative question, we do not report quantitative results. Instead, we qualitatively analyze the learned features and their hierarchical structure by examining the top activated features for a set of candidate tokens (e.g., "puppy"). We analyze these features in the same way as we do for the embeddings, examining the max-activating examples for each feature.

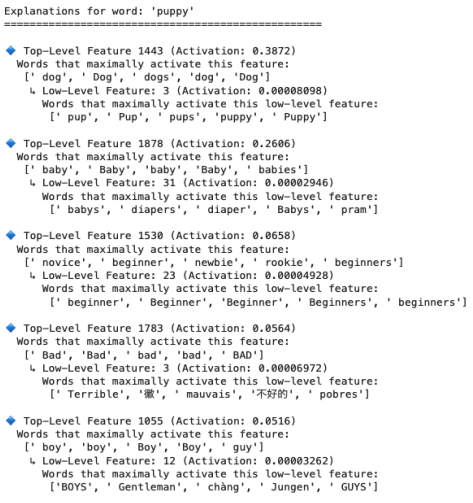
We test the orthogonality loss with and without the ℓ_1 regularizer, and find that the orthogonality loss does not seem to be necessary for interpretability and hierarchy. Nonetheless, we chose to keep it in our final runs on the embeddings as it helps reduce the number of dead latents, which is a practical concern.

We also test the ℓ_1 regularizer with and without the orthogonality loss, and find that it too does not seem to be necessary for interpretability and hierarchy. Ultimately, we chose to keep it for our final runs on the embeddings to allow for some adaptivity beyond the fixed top-k selection, as we have observed in practice that very low activations are often not meaningful, so we would like to encourage the model to use only the most informative features.

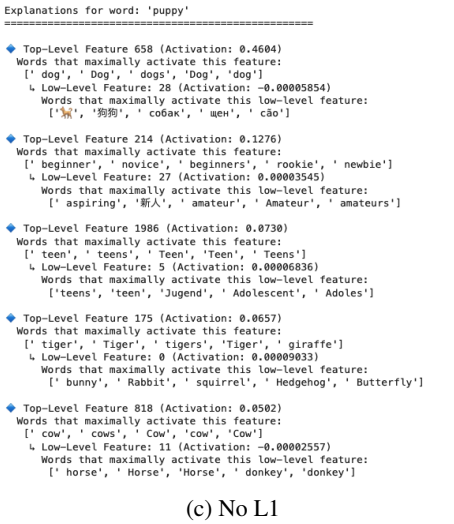
Displayed below in Figures 7, 8, 9, 10, 11, 12, and 13 are the example results of our ablations studies on the unembedding matrix. These were not chosen randomly but rather to show a variety of different types of features.



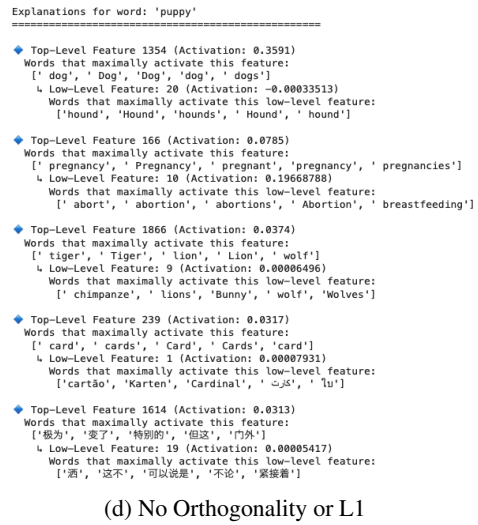
(a) Baseline



(b) No Orthogonality



(c) No L1

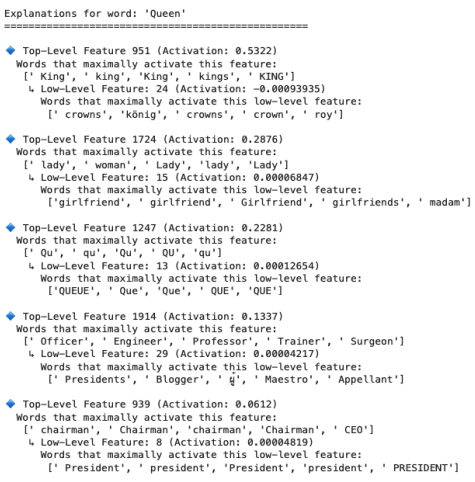


(d) No Orthogonality or L1

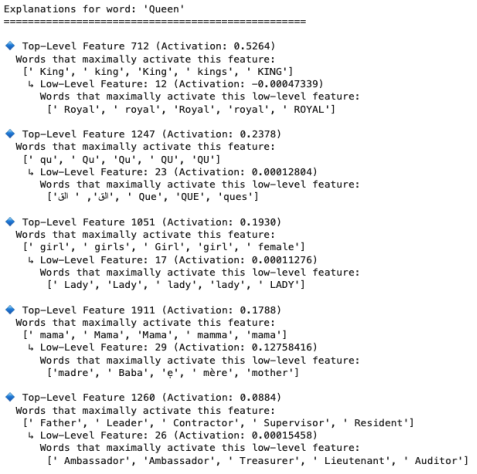
Figure 7: Ablation studies on the unembedding matrix show no obvious advantage from the orthogonality or ℓ_1 regularizers, though we chose to keep them for our final runs on the embeddings.

B.3 Importance of The Causal Inner Product

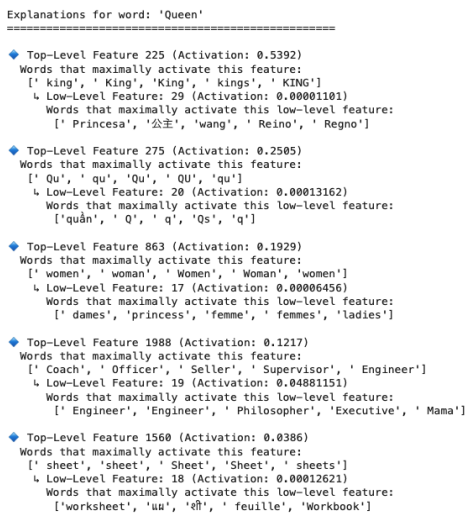
As briefly mentioned in our pre-processing section, there are theoretical reasons to believe that the Euclidean inner product does not align with the underlying geometry of the unembedding space. Our results below (in Figures 14, 15, 16, 17, 18, 19, and 20) validate this hypothesis, as we find that the unwhitened unembedding matrix does not yield meaningful features. The features are completely different from the baseline, and do not seem to have any coherent meaning, as illustrated below.



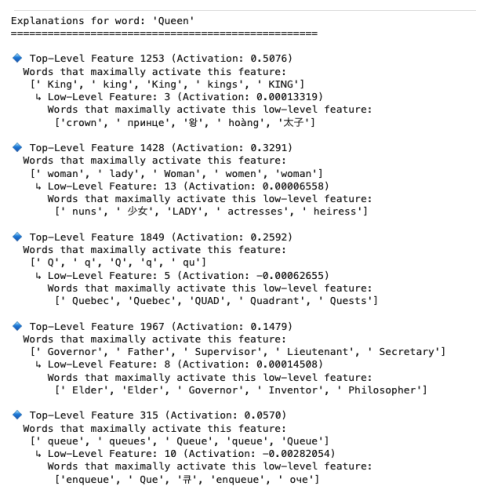
(a) Baseline



(b) No Orthogonality

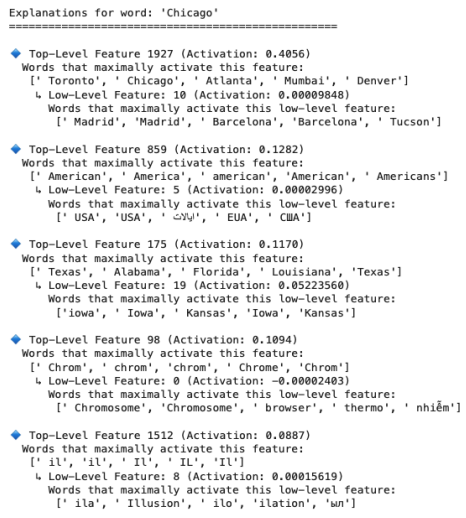


(c) No L1

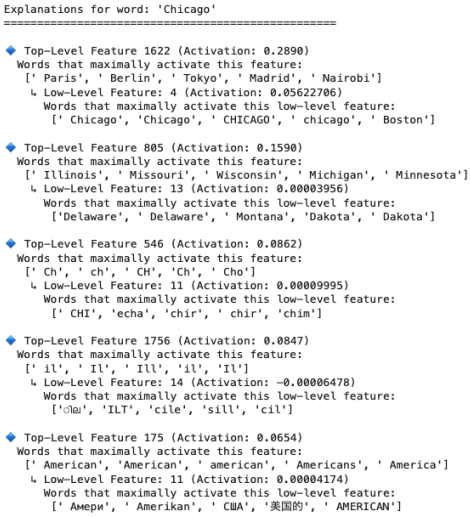


(d) No Orthogonality or L1

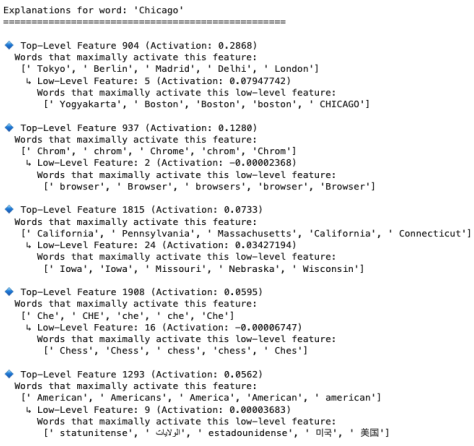
Figure 8: Ablation studies on the unembedding matrix show no obvious advantage from the orthogonality or ℓ_1 regularizers, though we chose to keep them for our final runs on the embeddings.



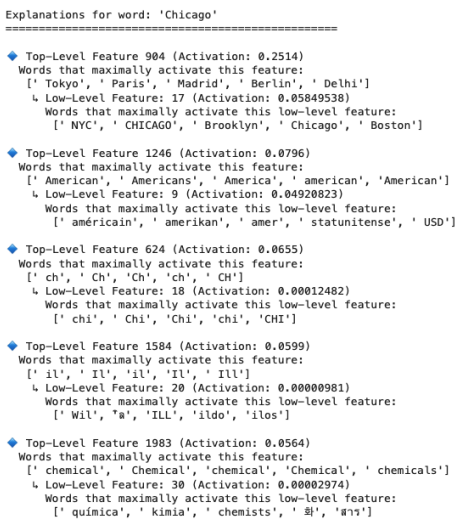
(a) Baseline



(b) No Orthogonality

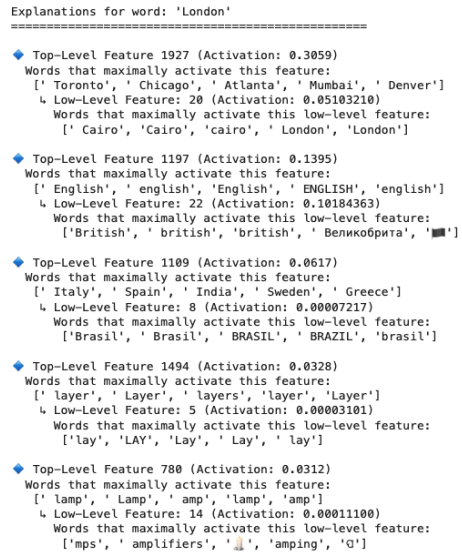


(c) No L1

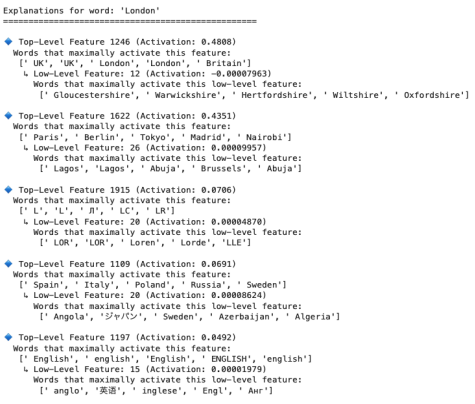


(d) No Orthogonality or L1

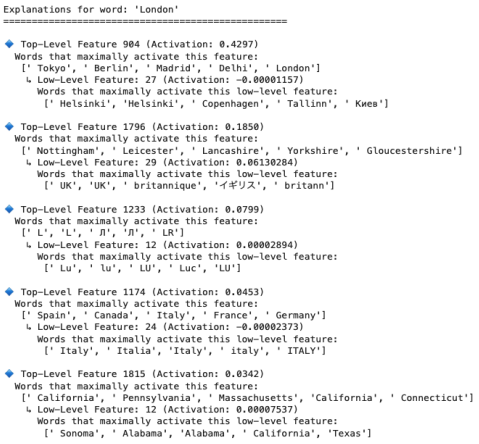
Figure 9: Ablation studies on the unembedding matrix show no obvious advantage from the orthogonality or ℓ_1 regularizers, though we chose to keep them for our final runs on the embeddings.



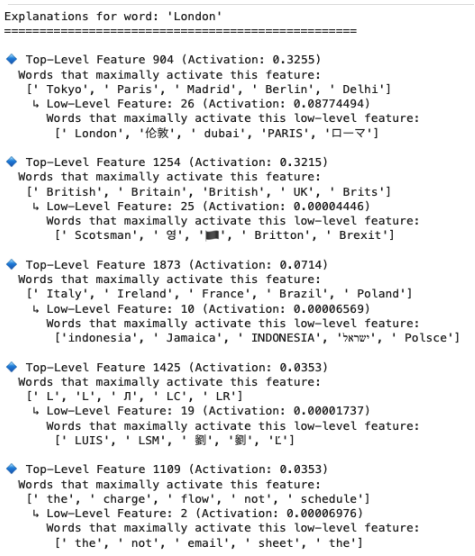
(a) Baseline



(b) No Orthogonality

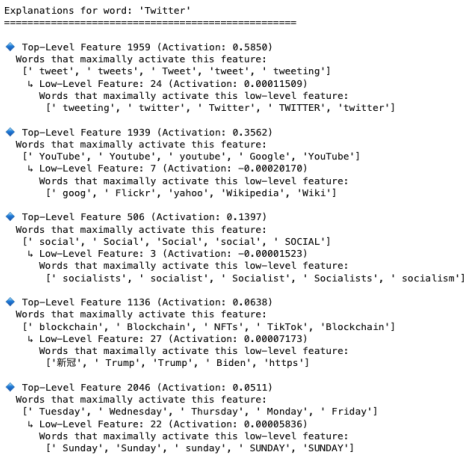


(c) No L1

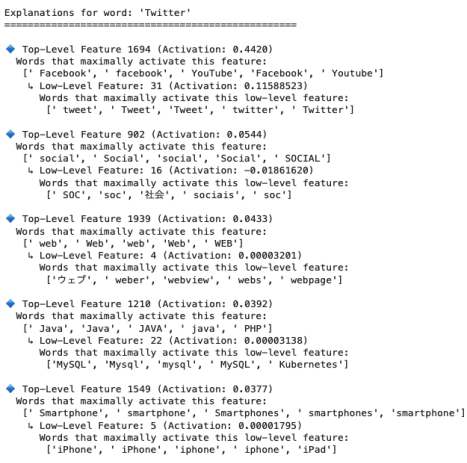


(d) No Orthogonality or L1

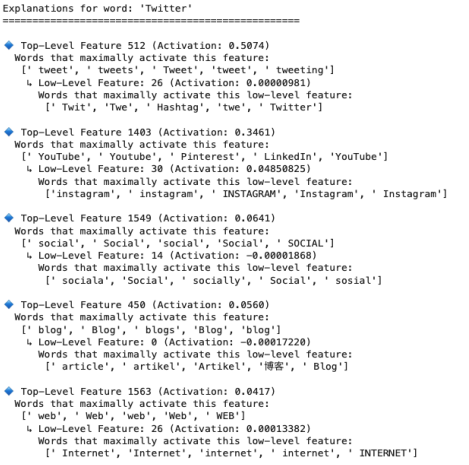
Figure 10: Ablation studies on the unembedding matrix show no obvious advantage from the orthogonality or ℓ_1 regularizers, though we chose to keep them for our final runs on the embeddings.



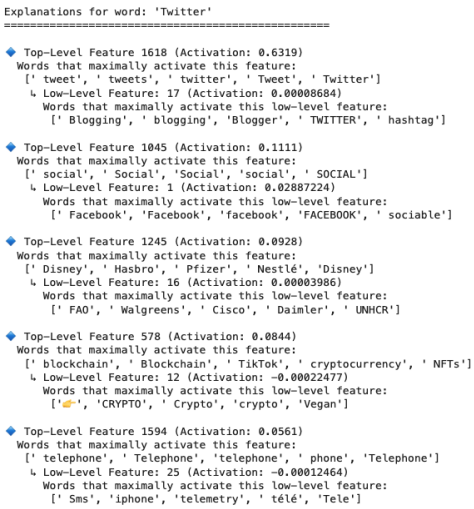
(a) Baseline



(b) No Orthogonality

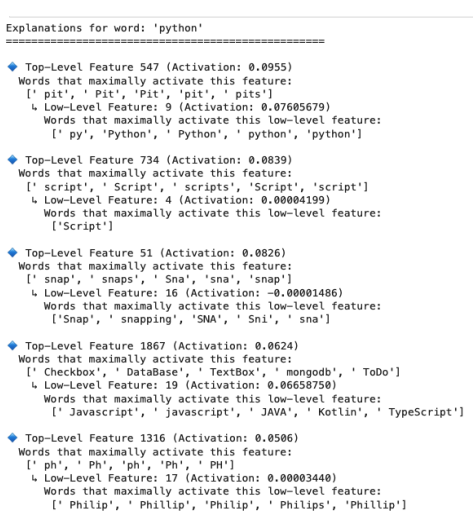


(c) No L1

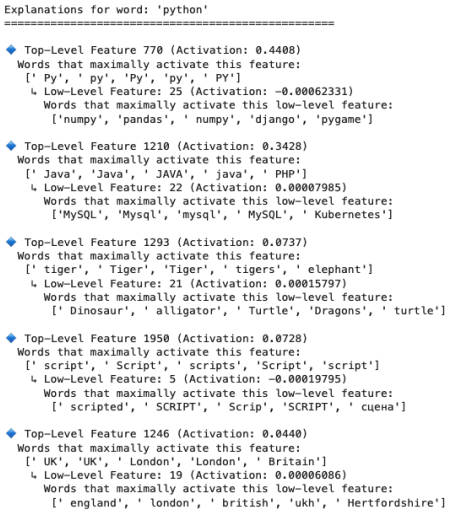


(d) No Orthogonality or L1

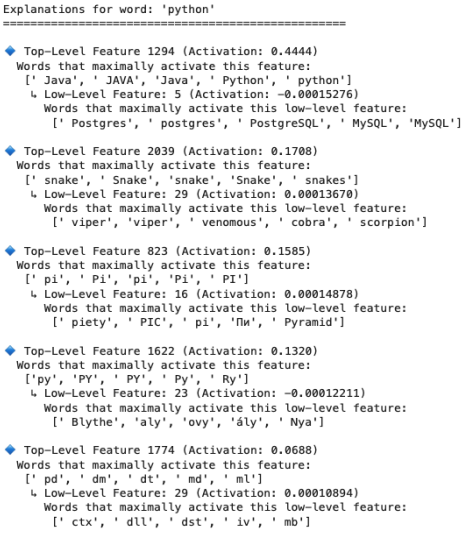
Figure 11: Ablation studies on the unembedding matrix show no obvious advantage from the orthogonality or ℓ_1 regularizers, though we chose to keep them for our final runs on the embeddings.



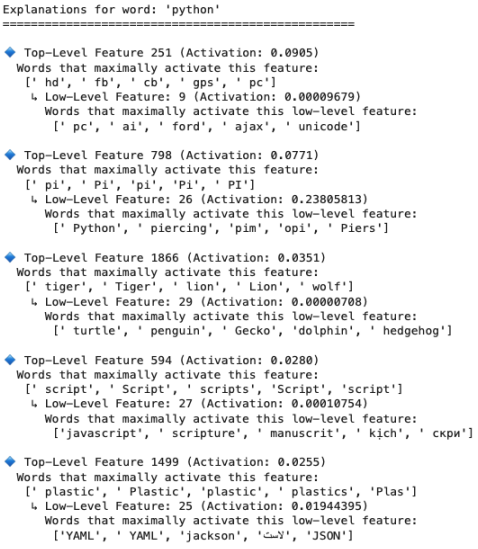
(a) Baseline



(b) No Orthogonality



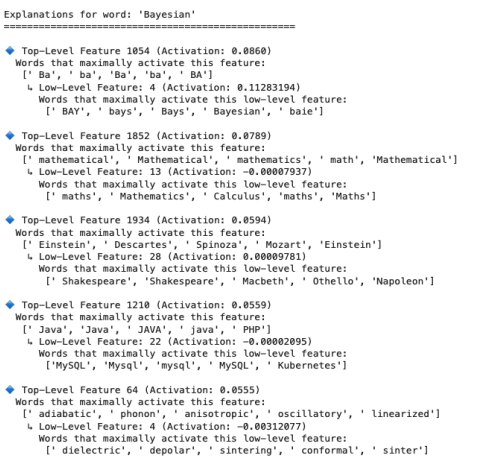
(c) No L1



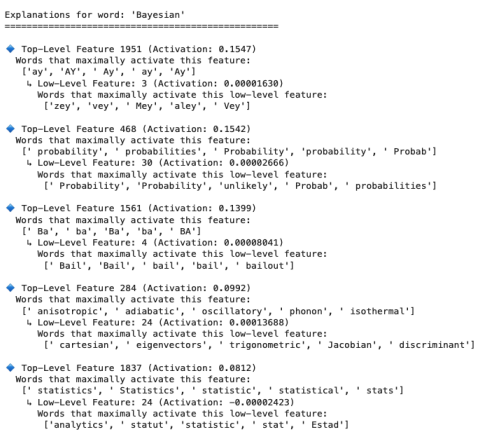
(d) No Orthogonality or L1

Figure 12: Ablation studies on the unembedding matrix show no obvious advantage from the orthogonality or ℓ_1 regularizers, though we chose to keep them for our final runs on the embeddings.

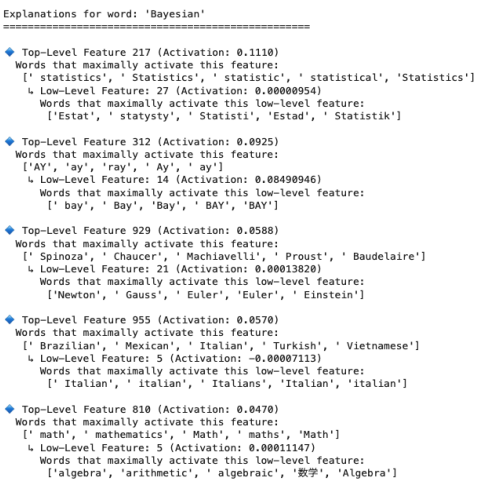
(a) Baseline



(b) No Orthogonality

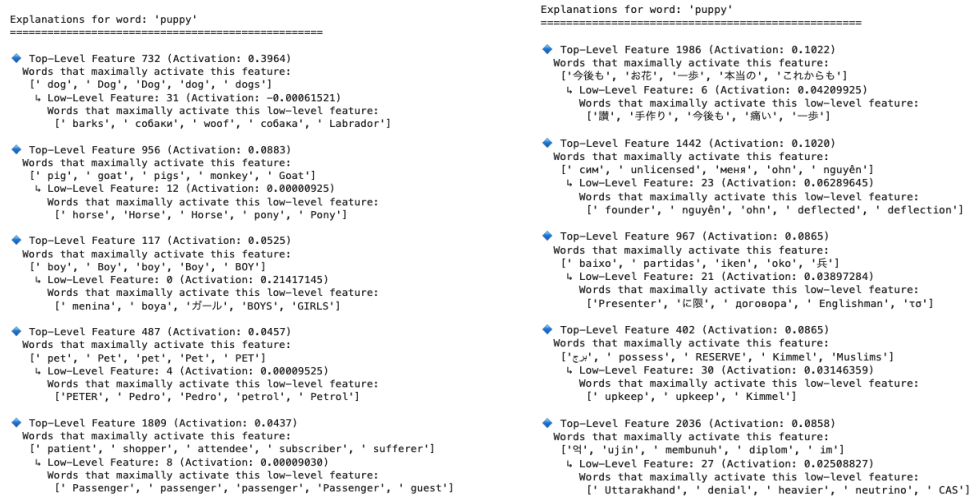


(c) No L1



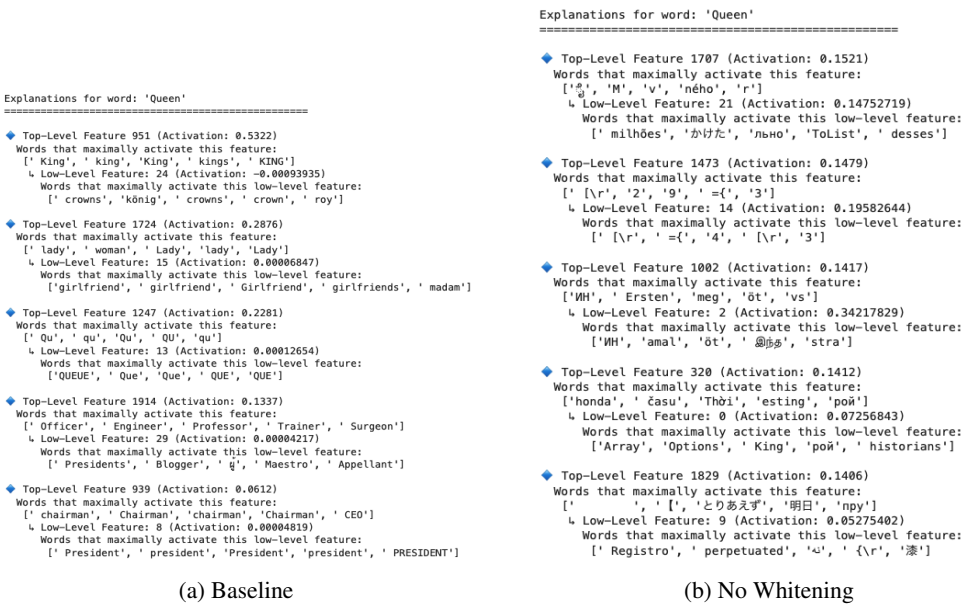
(d) No Orthogonality or L1

Figure 13: Ablation studies on the unembedding matrix show no obvious advantage from the orthogonality or ℓ_1 regularizers, though we chose to keep them for our final runs on the embeddings.



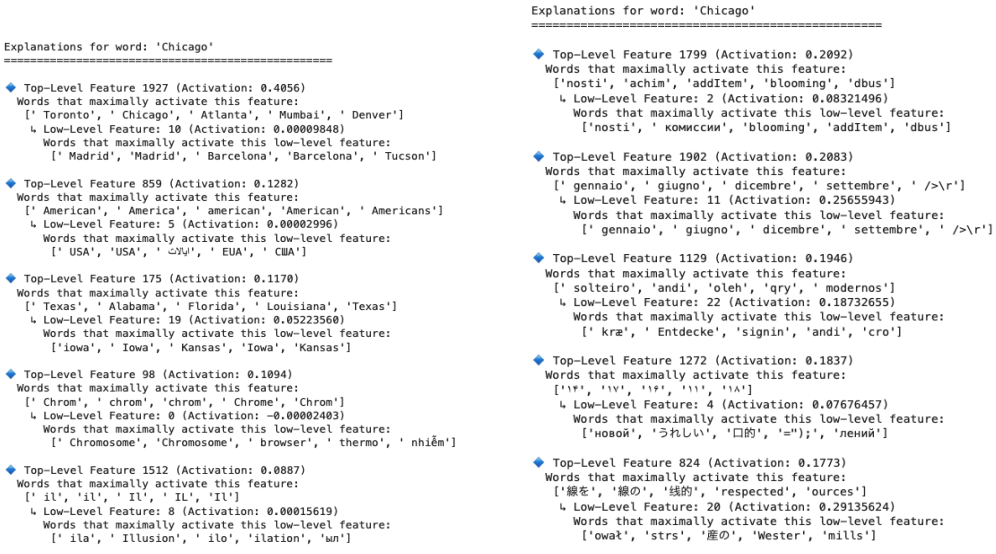
(a) Baseline (b) No Whitening

Figure 14: The causal inner product is necessary for the model to learn meaningful features.



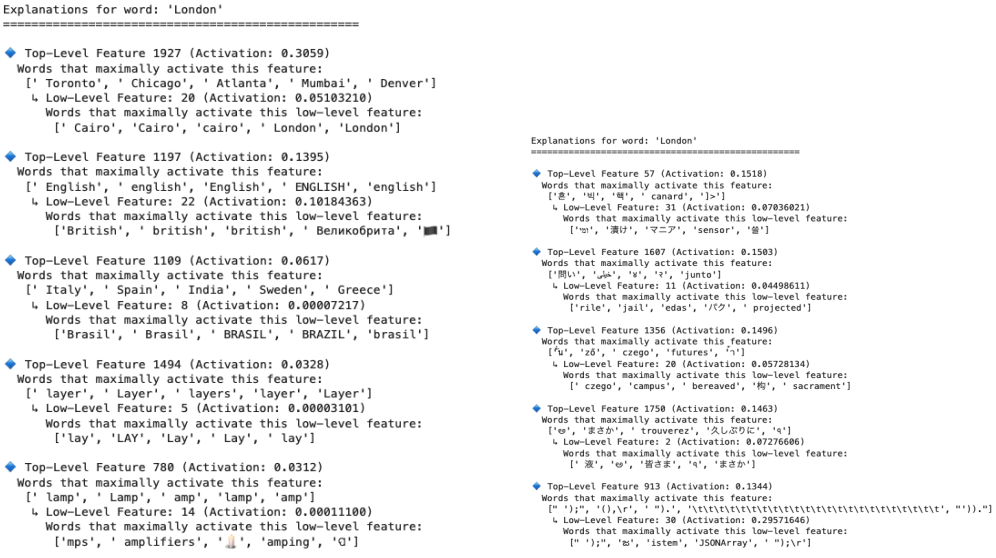
(a) Baseline (b) No Whitening

Figure 15: The causal inner product is necessary for the model to learn meaningful features.



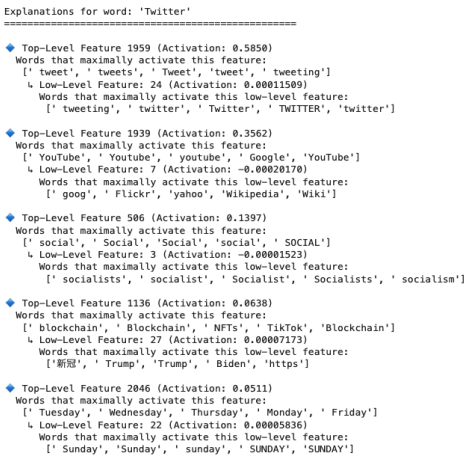
(a) Baseline (b) No Whitening

Figure 16: The causal inner product is necessary for the model to learn meaningful features.

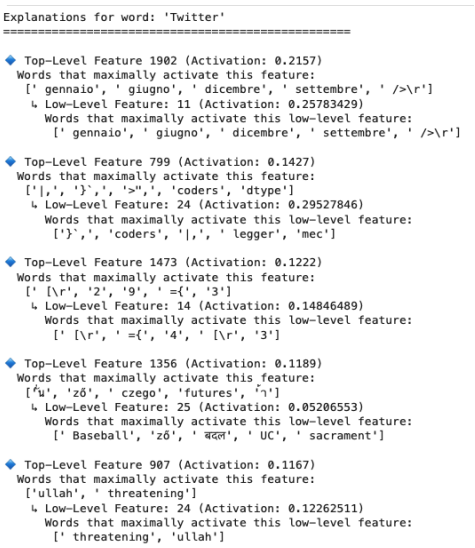


(a) Baseline (b) No Whitening

Figure 17: The causal inner product is necessary for the model to learn meaningful features.

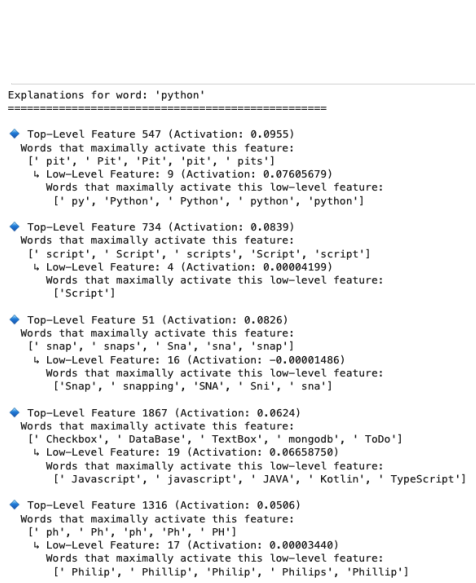


(a) Baseline

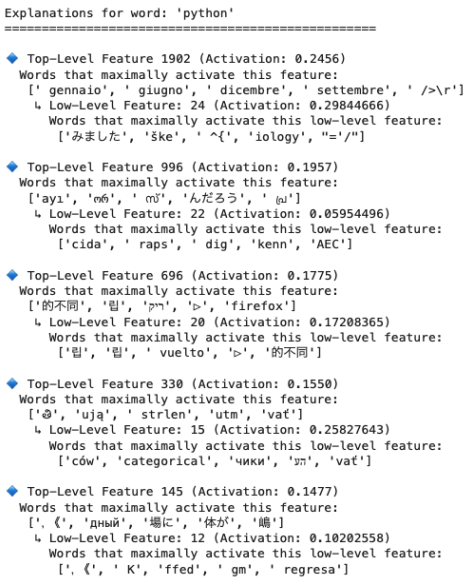


(b) No Whitening

Figure 18: The causal inner product is necessary for the model to learn meaningful features.

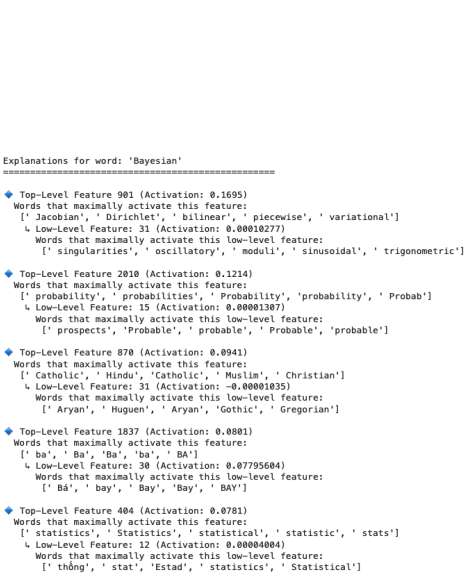


(a) Baseline

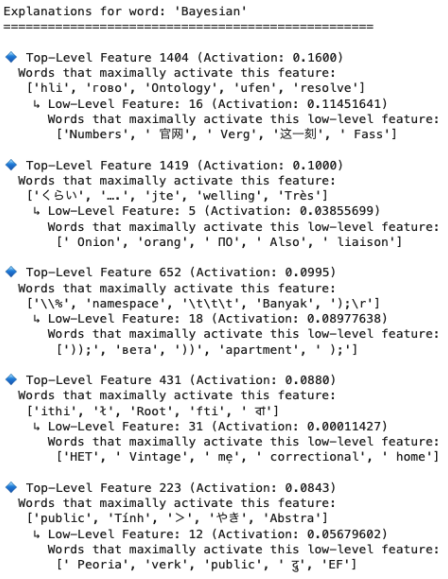


(b) No Whitening

Figure 19: The causal inner product is necessary for the model to learn meaningful features.



(a) Baseline



(b) No Whitening

Figure 20: The causal inner product is necessary for the model to learn meaningful features.

C MORE EXAMPLE FEATURES

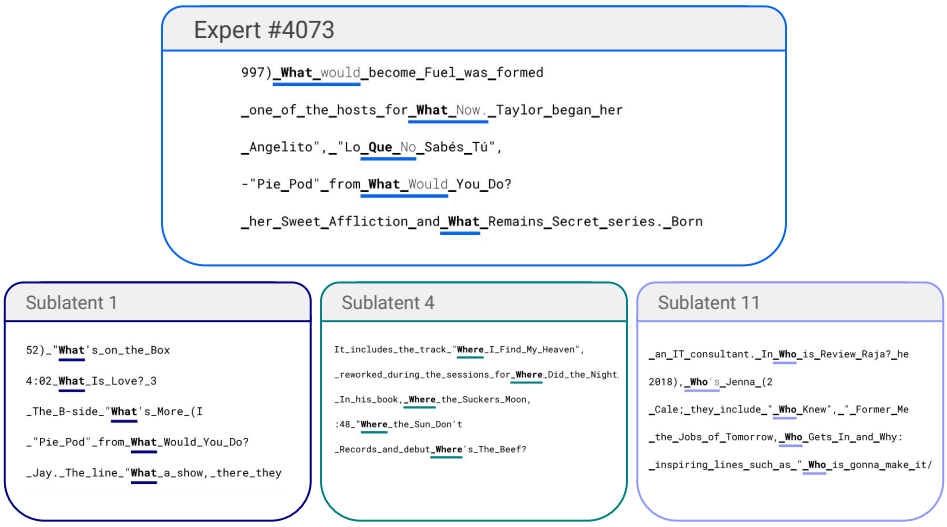


Figure 21: An example of a hierarchical feature from the 8K highlevel x 16 sublatent H-SAE model. A 'question-word' high level feature with "What", "Where" and "Who" sublatents.

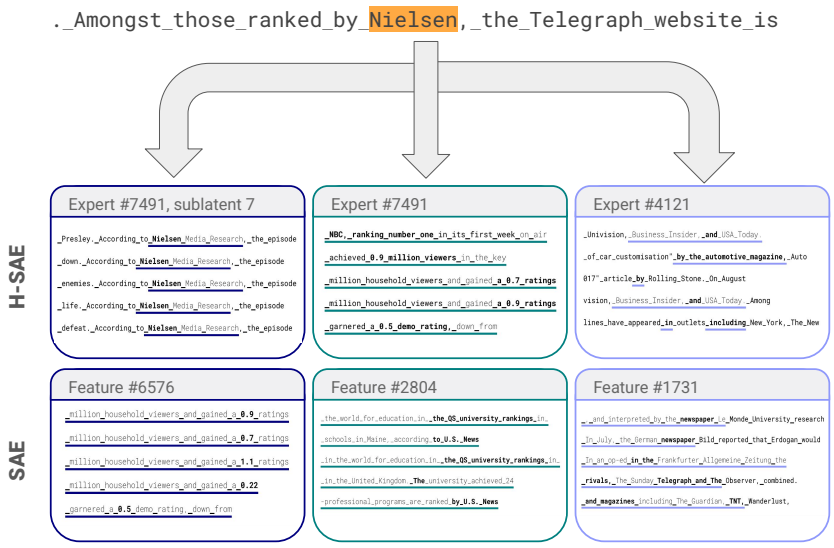


Figure 22: A comparison decomposition of the same context/embedding in the 8k x 16 H-SAE architecture and 8k standard SAE. Of note in the H-SAE is Expert #7491 which not only has the meaningful sublatent visualized here but also additional sublatents for various parts of writing about rating like demographics, specific ratings agencies like Nielsen, and mediums like TV episodes. Expert #4121 is a "magazines"/"news website" feature, promoting the following tokens heavily if added to the residual stream: _Forbes, _Bloomberg, _CNN, _BBC, _Daily, _Newsweek, _Reuters, _Huffington, _magazine, _The. We see a similar decomposition on the SAE, with a newspaper feature and "media ratings" feature. However, due to the lack of granular features, the specific ratings agency 'Nielsen' is not represented (rather the top-level ratings feature promotes the token 'Nielsen' more heavily in the SAE). The SAE uses a more generic "ratings"/"rankings" feature that shows signs of absorption with a "University Rankings" feature.

D COMPARISONS WITH MATROYSHKA-SAE

D.1 SYNTHETIC BENCHMARK

The M-SAE paper introduces a synthetic benchmark that consists of 20 features organized hierarchically into 11 "parent features" (which can co-activate) and 9 child features split evenly among 3 of the parents (so 8 "parent" features have no children). We adapt this to 28 overall, with 12 parents, and 16 children split among 4 parents. Following the Matrysoka paper, we train a M-SAE with 28 features. We also train a 12x4 H-SAE (notice that this model is misspecified for the data because we allow all parents to have children). The M-SAE paper evaluates the reconstruction by cosine similarity between the optimal alignment of the learned and ground truth features. We find the M-SAE achieves a mean cosine similarity of 0.502 and the H-SAE 0.526 (despite the H-SAE being misspecified)

D.2 UNEMBEDDING DECOMPOSITION

We train an Hierarchical-SAE and Matroyshka-SAE for 10,000 steps on the unembeddings transformed by the causal inner product. The H-SAE is a top-k=5, 2k x 32 model and the M-SAE is matched with top-k=10 and 5 groups out of 65k total features (i.e. both models have the same number of overall latents and can select the same number of latents per reconstruction). The features are fairly similar, with the H-SAE often finding sparser representations even within its top-k (by having near-zero activations on irrelevant features), finding high-level features the M-SAE does not, and fewer uninterpretable features. The lower maximum activation may also be an advantage of the H-SAE, as it spreads its activation weight out over more composable features. These factors also combine with the reconstruction loss and computational advantage the H-SAE has over both an M-SAE and standard SAE (e.g. the H-SAE computes 3% as many feature activations during the encoding stage as compared to the M-SAE). Comparison decompositions follow:

Decomposition of ‘ Chicago’

Feature Num		**Activation**		**Top Tokens**	
H-SAE	*M-SAE*	*H-SAE*	*M-SAE*	*H-SAE*	*M-SAE*
1927	1368	0.4056	0.5899	[' Toronto', ' Chicago', ' Atlanta', ' Mumbai', ' Denver']	[' Chicago', 'Chicago', ' CHICAGO', ' chicago', 'chicago']
1927.5	737	1E-3	0.4034	[' Madrid', 'Madrid', ' Barcelona', 'Barcelona', ' Tucson']	[' Seattle', ' Atlanta', ' Detroit', ' Denver', ' Chicago']
859	44230	0.1282	0.0462	[' American', ' America', ' american', 'American', ' Americans']	['chicago', 'ற்று', ' chicago', 'ற்ற', 'Chicago']
859.5	37392	3E-4	0.0164	[' USA', 'USA', ' ایالات', ' EUA', ' США']	[' Missouri', ' Wisconsin', ' Indianapolis', ' Kansas', ' Tennessee']
175	15282	0.1170	0.0403	[' Texas', ' Alabama', ' Florida', ' Louisiana', 'Texas']	[' Earth', ' West', ' South', ' King', ' Bay']
175.19	3961	0.0522	0.0525	['iowa', ' Iowa', ' Kansas', 'Iowa', 'Kansas']	[' Pennsylvania', ' Michigan', ' Illinois', ' Wisconsin', ' Ohio']
98	4981	0.1094	0.0329	[' Chrom', ' chrom', 'chrom', ' Chrome', 'Chrom']	['NYC', ' NY', ' NYC', 'NY', ' nyc']
98.0	34993	-2E-4	0.0069	[' Chromosome', 'Chromosome', ' browser', ' thermo', ' nhiễm']	[' MEXICO', 'mexico', ' mexico', 'Mexico', ' Mexico']
1512	64996	0.0887	0.0067	[' il', 'il', ' Il', ' IL', 'Il']	[' ới', ' giới', 'ời', 'Amore', ' }}{\\']
1512.8	34031	1.6E-3	0.0066	[' ila', ' Illusion', ' ilo', 'ilation', 'ыл']	[' rör', ' haberse', ' haber', ' skydd', 'cc']

Decomposition of ‘ puppy’

Feature Num		**Activation**		**Top Tokens**	
H-SAE	*M-SAE*	*H-SAE*	*M-SAE*	*H-SAE*	*M-SAE*
732	282	0.3964	0.6628	[' dog', ' Dog', 'Dog', 'dog', ' dogs']	[' puppy', ' Puppy', 'Puppy', 'puppy', ' puppies']
732.31	177	-0.0006	0.3691	[' barks', ' собаки', ' woof', ' собака', ' Labrador']	[' dog', ' dogs', ' Dog', 'dog', 'Dog']
956	27992	0.0883	0.0356	[' pig', ' goat', ' pigs', ' monkey', ' Goat']	[' brother', ' mother', ' father', ' sister', ' son']
956.12	3008	9E-5	0.0633	[' horse', 'Horse', ' Horse', ' pony', ' Pony']	[' infant', ' Infant', ' infants', ' baby', ' babies']
117	13729	0.0525	0.0361	[' boy', ' Boy', 'boy', 'Boy', ' BOY']	[' wood', ' fish', ' glass', ' milk', ' snow']
117.0	59925	0.2141	0.0068	[' menina', ' boya', 'ガール', 'BOYS', 'GIRLS']	['\uf075', ' \uf075', ' zamanda', '玩笑', '\uf06e']
487	46772	0.0457	0.0067	[' pet', ' Pet', 'pet', 'Pet', ' PET']	['เยอะ', 'มากๆ', 'ะ', ' oxpa', 'เขียน']
487.4	6188	1E-4	0.0264	['PETER', ' Pedro', 'Pedro', 'petrol', ' Petrol']	[' Teddy', 'Teddy', 'teddy', ' teddy', ' Ted']
1809	36597	0.0437	0.0065	[' patient', ' shopper', ' attendee', ' subscriber', ' sufferer']	['๋ด', 'ふる', 'จด', 'inau', '獣']
1809.8	63546	9E-4	0.0065	[' Passenger', ' passenger', 'passenger', 'Passenger', ' guest']	['MDL', ' MDL', 'mdl', 'NOV', 'MRP']

Decomposition of ‘ Queen’

Feature Num		**Activation**		**Top Tokens**	
H-SAE	*M-SAE*	*H-SAE*	*M-SAE*	*H-SAE*	*M-SAE*
951	99	0.5322	0.8691	[' King', ' king', 'King', ' kings', ' KING']	[' Queen', 'Queen', ' queen', ' QUEEN', ' queens']
951.24	2304	-9E-3	0.1364	[' crowns', 'könig', ' crowns', ' crown', ' roy']	[' Coach', ' Owner', ' Officer', ' Manager', ' Administrator']
1724	15282	0.2876	0.0816	[' lady', ' woman', ' Lady', 'lady', 'Lady']	[' Earth', ' West', ' South', ' King', ' Bay']
1724.15	2304	7E-4	0.0109	['girlfriend', ' girlfriend', ' Girlfriend', ' girlfriends', ' madam']	[' Staten', '諫', 'Axes', ' Rhode', 'Rhode']
1247	43216	0.2281	0.0076	[' Qu', ' qu', 'Qu', ' QU', 'qu']	['goers', ' Watcher', 'qc', 'QC', ' VIEWS']
1247.13	30719	1.3E-3	0.0147	['QUEUE', ' Que', 'Que', ' QUE', 'QUE']	['Jerusalem', ' Jerusalem', ' القدس', 'lande', ' Haifa']
1914	53291	0.1337	0.0073	[' Officer', ' Engineer', ' Professor', ' Trainer', ' Surgeon']	['미', ' 미', 'ミ', ' ミ', '숏']
1914.29	34322	4E-4	0.0073	[' Presidents', ' Blogger', ' 엇', ' Maestro', ' Appellant']	['寻求', ' recibido', ' kuitenkin', 'medal', '淘']
939	45011	0.0612	0.0073	[' chairman', ' Chairman', 'chairman', 'Chairman', ' CEO']	[' 伸', '伸', ' Bronson', '肘', 'レッド']
939.8	59557	5E-3	0.0071	[' President', ' president', 'President', 'president', ' PRESIDENT']	[' VTT', ' Norsk', ' Matti', ' formant', ' Marat']

Decomposition of ‘ Bayesian’

Feature Num		**Activation**		**Top Tokens**	
H-SAE	*M-SAE*	*H-SAE*	*M-SAE*	*H-SAE*	*M-SAE*
901	8866	0.1695	0.4217	[' Jacobian', ' Dirichlet', ' bilinear', ' piecewise', ' variational']	['Bayesian', ' Bayesian', ' Bayes', 'Bayes', ' bayonet']
901.32	9192	1E-3	0.1400	[' singularities', ' oscillatory', ' moduli', ' sinusoidal', ' trigonometric']	['Probability', ' Probability', 'probability', ' probability', '概率']
2010	2552	0.1214	0.2152	[' probability', ' probabilities', ' Probability', 'probability', ' Probab']	[' Dirichlet', ' Jacobian', ' Cauchy', ' Laplace', ' Poincaré']
2010.15	3286	1E-4	0.1930	[' prospects', 'Probable', ' probable', ' Probable', 'probable']	['ay', 'AY', ' Ay', ' ay', 'Ay']
870	858	0.0941	0.1309	[' Catholic', ' Hindu', 'Catholic', ' Muslim', ' Christian']	[' ba', ' Ba', 'Ba', 'ba', ' BA']
870.31	1395	-1E-4	0.686	[' Aryan', ' Huguen', ' Aryan', 'Gothic', ' Gregorian']	[' probability', ' likelihood', ' chances', ' probabilities', ' Probability']
1837	1367	0.0801	0.0593	[' ba', ' Ba', 'Ba', 'ba', ' BA']	[' Catholic', 'Catholic', ' Catholics', ' catholic', ' Muslim']
1837.30	5299	0.0780	0.0459	[' Bá', ' bay', ' Bay', 'Bay', ' BAY']	[' statistics', ' statistical', ' Statistics', 'Statistics', 'statistics']
404	57380	0.0781	0.0099	[' statistics', ' Statistics', ' statistical', ' statistic', ' stats']	['広場', ' priors', '样本', 'Bayesian', ' YC']
404.12	39412	4E-4	0.0085	[' thống', ' stat', 'Estad', ' statistics', ' Statistical']	[' epistem', ' methodological', ' Epis', ' EPISODE', ' evangel']