Leveraging the Sequential Nature of Language for Interpretability

Usha Bhalla [*1,2] **Alex Oesterling** [*1] **Claudio Mayrink Verdun** [1] **Flavio P. Calmon** [1] **Himabindu Lakkaraju** [1]

Abstract

Interpretability strives to discover the concepts learned and represented by models, frequently with unsupervised learning methods such as dictionary learning. While such methods allow for flexibility and applicability to a wide variety of domains, recent implementations such as sparse autoencoders (SAEs) discard valuable modality-specific information and priors that we have on the structure of different data modalities. In this work, we argue that the temporal dimension of language is a rich feature source that can be leveraged by dictionary learning methods in a self-supervised manner, allowing for better learning and disentanglement of semantic and syntactical features represented by language models. We propose a data-generating process for such features, which informs a novel approach to train Temporal SAEs that can extract semantic concepts from natural language. We experimentally verify that accounting for the temporal structure of language improves SAEs' ability to capture semantic features in text data with minimal loss in performance.

1. Introduction

The field of machine learning interpretability seeks to understand what information models represent and encode – often with goals such as being able to rigorously audit models, control and steer them, or learn something new about the data itself or the functions and algorithms the model has learned. In cases where one has a specific concept of interest that they are trying to understand or control, they can leverage supervised interpretability methods such as probing (Köhn, 2016; Alain & Bengio, 2016; Belinkov, 2022) or steering (Subramani et al., 2022); however, many other cases exist where we may want to discover concepts in an *unsupervised* manner, to better understand and learn about the variety of features that models rely on. Dictionary learning (Dumitrescu & Irofti, 2018; Bricken et al., 2023), and more specifically, sparse autoencoders (Ng et al., 2011; Makhzani & Frey, 2013), have emerged as the primary method to perform such unsupervised concept discovery, with prior works finding they discover, human-interpretable and sometimes even unknown features that can be used to steer models. In fact, SAEs have been applied across a variety of modalities: language, vision, x-rays (Abdulaal et al., 2024), protein sequencing (Garcia & Ansuini, 2025), time-series data (Wu et al., 2019), and more, with almost no changes or adaptations to the architecture for these different data types. However, this standard, out-of-the-box application across modalities assumes no underlying structure of the data. While this lack of supervision allows for high flexibility, it forgoes any priors and inductive biases we have about the data and the concepts we hope that SAEs will recover.

In this work, we focus on natural language and demonstrate how we can leverage intuitions and priors related to its temporal behavior to improve SAEs. To do so, we first construct a potential data-generating process (DGP) for language data that accounts for its sequential nature. In particular, we propose that high-level, abstract features should be *smooth* over time, and that low-level, syntactic features should be independent from the high-level features. We further expect that model representations encode a mixture of features relating to the semantics of the text, the prior context, and the syntactic requirements for the next-token generation task. We then propose a novel dictionary learning procedure that accounts for these assumptions and validate its efficacy through explorations of the ability of SAEs to recover the semantic, sequential, and syntactic features of the data.

Our contributions are the following:

1. We describe a simple data-generating process to model sequential structure in language.
2. We propose a novel loss function to ensure temporal consistency in SAE features.
3. We demonstrate empirically that Temporal SAEs exhibit better semantic structure while maintaining competitive performance to state-of-the-art SAEs.

[*]Equal contribution [1]Harvard University [2]Kempner Institute. Correspondence to: Usha Bhalla <usha_bhalla@g.harvard.edu>, Alex Oesterling <aoesterling@g.harvard.edu>.

ICML 2025 Workshop on Assessing World Models.

2. Related Work

Sparse Autoencoders. In recent years, SAEs have emerged as a popular mechanistic interpretability technique for self-supervised concept discovery. They aim to explain models by decomposing intermediate model activations into sparse, human-interpretable feature spaces. While they were initially promising for addressing the problem of polysemanticity, where a single neuron in a model can represent multiple features (Elhage et al., 2022), in practice they have been shown to create new problems, such as feature splitting and absorption (Chanin et al., 2024), where features are split across multiple features or absorbed into less interpretable sub-features. To address these subsequent issues, methods such as Matryoshka SAEs (Bussmann et al., 2025) and transcoders (Paulo et al., 2025) have been proposed, which learn hierarchical and causal features. Recent work has also proposed learning dictionary features that are constrained to the data manifold (Fel et al., 2025) and reflect intuition about the geometry of model latent spaces (Hindupur et al., 2025), allowing for the recovery of heterogeneous concepts. However, all of these works assume a fully unsupervised objective for learning SAEs, treating each token in the training data as i.i.d., without acknowledging the temporal aspect of language and other sequential modalities.

Cognitive and Computational Models of Language. Literature in cognitive science has long studied the difference between syntactical and semantic content in language, with empirical evidence of different developmental trends for the two during human language acquisition (Brown, 1973) as well as differences in patterns of brain activity for both (Neville et al., 1992). Statistical methods subsequently demonstrated the ability to discover semantic content from language via topic modeling such as with Latent Dirichlet Allocation (Blei et al., 2003) and syntactic content with distributional methods (Redington et al., 1998) and Hidden Markov Models (Manning & Schutze, 1999). (Griffiths et al., 2004) combine computational models of semantics and syntax into HMM-LDA to model both simultaneously. Importantly, they argue that semantics in language exhibit long-range behavior, with different words or sentences in the same document having similar semantic content, whereas syntax is mostly dependent on short-range interactions. This perspective informs our model of the data-generating process of model latents and our training approach for inducing temporal consistency in SAEs.

3. Data-Generating Process

We formalize our data-generating process as such. Consider a speaker who is producing language, or a sequence of tokens $\tau_1, ..., \tau_T$. When the speaker produces each token τ_t, they take into account many factors — their intent in speaking, the prior context of the token (i.e. what has already been said), syntactic requirements, and other implicit features corresponding to speaker idiosyncrasies (such as their accent, their method of language production, or linguistic style). These factors can be modeled as latent variables that control the language generation process, and they can be generally categorized into two types: variables that encode *high-level or global* information, $\mathbf{h}_t$, and variables that encode *low-level or local* information $\mathbf{l}_t$. High-level variables can be thought of as features that are invariant to the specific token, such as those capturing semantics and intent. Conversely, low-level information pertains to the specific timestep or token being produced, such as a word's grammatical gender.

We model the speaker's language production process as a randomized function mapping the context and these latent variables to the next token

$$\tau_t = \phi(\tau^{t-1}, \mathbf{h}_t, \mathbf{l}_t),$$

where τ^{t-1} represents the previously-uttered tokens $\tau_1, ..., \tau_{t-1}$. Given a language model M, we pass tokens τ^T into M which produces latent vectors $\{\mathbf{x}_t^L\}_{t=1}^T \in \mathbb{R}^d$ at layer L. For simplicity, we analyze a single layer and drop the L superscript.

Our goal is to recover M's encoding of the data-generating latent variables by decomposing its representations into interpretable features corresponding to $\mathbf{h}_t, \mathbf{l}_t$. We ground our DGP in computational cognitive science and linguistics literature, where statistical models of language found success with similar assumptions (Griffiths et al., 2004). We specify the assumptions of our DGP, specifically regarding the nature of $\mathbf{h}_t, \mathbf{l}_t$, below.

1. $\mathbf{h}_t$ is time invariant, meaning two tokens $\mathbf{x}_t, \mathbf{x}_{t'}$ sampled from the same sequence should have similar latents $\mathbf{h}_t \approx \mathbf{h}_{t'}$. Essentially, for a given sequence of text, the semantics should be relatively constant over time.

2. In a language model, $\mathbf{h}_t$ and $\mathbf{l}_t$ are represented *additively*: we can decompose $\mathbf{x}_t$ as $\mathbf{x}_t^{\mathbf{h}} + \mathbf{x}_t^{\mathbf{l}}$ where $\mathbf{x}_t^{\mathbf{h}}$ captures the high level features used to generate $\mathbf{x}_t$ and similarly for $\mathbf{x}_t^{\mathbf{l}}$.

4. Temporal Sparse Autoencoders

We propose a modification to existing SAE architectures that leverages the sequential nature of language data as follows. Sparse autoencoders are comprised of an encoder, decoder, and nonlinear activation function. We decompose the encoders and decoders into two parts each: one that represents the high-level features $\mathbf{h}_t$ present in the input data $\mathbf{x}_t$ and one that represents the low-level features $\mathbf{l}_t$.

We partition the SAE feature space into high level and low level features. Without loss of generality we assume the first h indices are our high level features and the last $m-h$ indices are our low level features, where m is the number of features in the SAE. The SAE architecture can be defined as the following, taking in input $\mathbf{x}_t \in \mathbb{R}^d$:

$$\mathbf{f}(\mathbf{x}_t) = \sigma(\mathbf{W}^{\text{enc}}\mathbf{x}_t + \mathbf{b}^{\text{enc}}),$$
$$\hat{x}(\mathbf{f}) = \mathbf{W}^{\text{dec}}\mathbf{f}(\mathbf{x}_t) + \mathbf{b}^{\text{dec}}.$$

Here, $\mathbf{W}^{\text{enc}} \in \mathbb{R}^{m\times d}$ is the encoder matrix, and $\mathbf{W}^{\text{dec}} \in \mathbb{R}^{d\times m}$ is the decoder comprised of high-level features $\mathbf{W}^{\text{dec}}_{0:h} \in \mathbb{R}^{d\times h}$ and low-level features $\mathbf{W}^{\text{dec}}_{h:m} \in \mathbb{R}^{d\times(m-h)}$ such that their concatenation equals $\mathbf{W}^{\text{dec}}$. $\mathbf{b}^{\text{enc}} \in \mathbb{R}^d$ and $\mathbf{b}^{\text{dec}} \in \mathbb{R}^d$ are the encoder and decoder bias respectively. We define the following loss function, where the high-level features $\mathbf{f}_{0:h}(\mathbf{x}_t)$ should reconstruct the input and the low-level features $\mathbf{f}_{h:m}(\mathbf{x}_t)$ should reconstruct the residual, similar to the Matryoshka SAE objective in (Bussmann et al., 2025).

$$\mathcal{L}(\mathbf{x}_t) = \mathcal{L}_H + \mathcal{L}_L + \alpha\mathcal{L}_{\text{contr}},$$
$$\mathcal{L}_H = \|\mathbf{x}_t - \mathbf{W}^{\text{dec}}_{0:h}\mathbf{f}_{0:h}(\mathbf{x}_t) + \mathbf{b}^{\text{dec}}\|_2^2,$$
$$\mathcal{L}_L = \|\mathbf{x}_t - \mathbf{W}^{\text{dec}}\mathbf{f}(\mathbf{x}_t) + \mathbf{b}^{\text{dec}}\|_2^2.$$

We then add a training objective that encourages $\mathbf{W}^{\text{enc}}_{0:h}$ to learn features that respect our assumptions about $\mathbf{h}_t$. We do this by adding a contrastive term to the loss function that encourages $\mathbf{W}^{\text{enc}}_{0:h}\mathbf{x}_t$ to be similar to $\mathbf{W}^{\text{enc}}_{0:h}\mathbf{x}_{t-1}$, as we expect high-level features to be similar for two tokens from the same sequence, especially for two adjacent tokens. Let $\mathbf{z}_t$ be the high-level features $\mathbf{f}_{0:h}(\mathbf{x}_t)$, and let $s(\mathbf{x},\mathbf{y})$ be the cosine similarity between vectors $\mathbf{x}$ and $\mathbf{y}$ in the same latent space. Our contrastive loss at temperature λ is defined as

$$\mathcal{L}_{\text{contr}} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{s(\mathbf{z}_t^{(i)},\mathbf{z}_{t-1}^{(i)})/\lambda}}{\sum_{j=1}^{N}e^{s(\mathbf{z}_t^{(i)},\mathbf{z}_{t-1}^{(j)})/\lambda}} - \frac{1}{N}\sum_{j=1}^{N}\log\frac{e^{s(\mathbf{z}_{t-1}^{(j)},\mathbf{z}_t^{(j)})/\lambda}}{\sum_{i=1}^{N}e^{s(\mathbf{z}_{t-1}^{(i)},\mathbf{z}_t^{(j)})/\lambda}},$$

where N is our batch size and $\mathbf{z}_t^{(i)}$ is the ith latent vector in the batch. In practice, we load activations in pairs $\mathbf{x}_t, \mathbf{x}_{t-1}$ and shuffle the pairs to get diversity in each batch. We additionally explore sampling the second token uniformly over past tokens $\mathbf{x}_1, ..., \mathbf{x}_{t-1}$ to encourage long range semantic consistency (Appendix A.1.1). Intuitively, we expect that for high-level, abstract, and semantic features, tokens from the same sequence should have more similar activations than those sampled from two sequences randomly. Thus, we constrain the high-level dictionary to represent these as close in SAE feature space. For low-level, token-specific features, we do not have the same expectation, but by nature of fitting the residual left over by the high-level component of the network, our loss naturally encourages the remaining latents to capture higher-variance features over a sequence.

Table 1. Performance metrics for BatchTopK, Matryoshka, and Temporal SAEs for Pythia-160m and Gemma2-2b.

	SAE Model	FVU	% Dead	$\mathcal{L}_{L_2}$	Cos. Sim.
Pythia	Temporal	0.09	34%	10.4	0.89
	Matryoshka	0.07	44%	9.2	0.91
	BatchTopK	0.07	62%	9.3	0.91
Gemma	Temporal	0.41	1%	131	0.86
	Matryoshka	0.38	1%	119	0.88
	BatchTopK	0.38	1%	122	0.88

5. Experiments

We conduct experiments on Pythia 160M (Biderman et al., 2023) and Gemma2-2b (Team et al., 2024) to understand whether Temporal SAEs learn clear high- and low-level features, and whether these features capture the types of concepts we expect. We compare against baselines of BatchTopK SAEs (Bussmann et al., 2024) and Matryoshka SAEs (Bussmann et al., 2025).

5.1. Implementation Details

We train and evaluate SAEs on the residual streams of Gemma2-2b (layer 18, 16k features) and Pythia-160m (layer 8, 32k features) on the `RedPajama-Data-1T-Sample` dataset (Weber et al., 2024). When training Temporal and Matryoshka SAEs, we split the dictionaries into two chunks, with the first being 20% of features for Pythia and 10% for Gemma. We only apply our temporal loss to the first chunk. All models were trained with the BatchTopK activation function (Bussmann et al., 2024) and auxiliary loss from (Gao et al., 2024), which encourages dead latents to reconstruct the error left from existing reconstructions. SAEs for Pythia were trained with a $k-$sparsity of 20 and Gemma SAEs were trained with $k = 40$. We compare our Temporal SAEs to baselines of Matryoshka and regular BatchTopK SAEs.

5.2. SAE Evaluation Metrics

We evaluate all SAEs on a 128k token sample from the Pile (Gao et al., 2020) on the following standard metrics to ensure that temporal consistency does not significantly degrade reconstruction quality.

Fraction of variance unexplained (FVU) measures the proportion of the variance of the model latent, $\mathbf{x}$, that the SAE reconstruction, $\hat{\mathbf{x}}$, fails to account for as $1-(\text{Var}(\mathbf{x}-\hat{\mathbf{x}})/\text{Var}(\mathbf{x}))$, where Var is the empirical variance. **$\mathbf{L}_2$ loss** measures the mean-squared error of the reconstruction, or

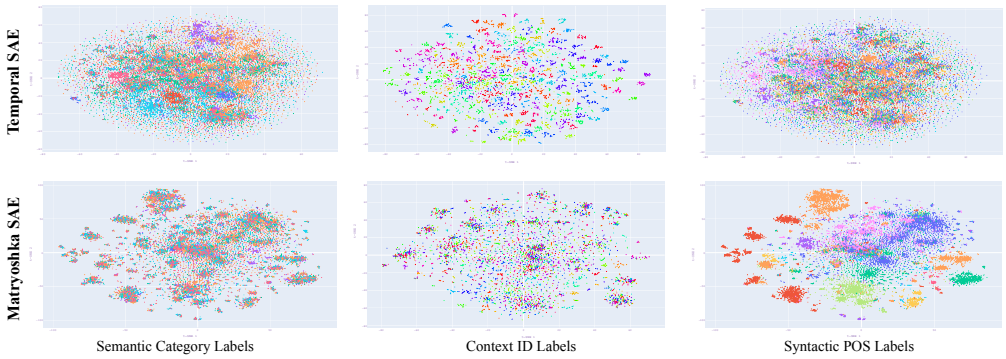

Figure 1. TSNE plots of the SAE feature activation for Temporal (top row) and Matryoshka (bottom row) SAEs. We label points by their semantic category (left), the context or sequence the token comes from (middle), and the part of speech of the token (right). We find that Temporal SAEs learn a balance of both semantic and syntactic information, whereas Matryoshka SAEs prioritize only syntactic information at the cost of all semantics.

$\|\mathbf{x} - \hat{\mathbf{x}}\|_2^2$. **Cosine similarity** computes the similarity of the reconstruction and SAE input as $\langle \mathbf{x}, \hat{\mathbf{x}} \rangle / (\|\mathbf{x}\|_2 \|\hat{\mathbf{x}}\|_2)$. Finally, the **percentage of dead latents** counts the percentage of SAE latent features that did not activate for any token in the evaluation dataset. We find that Temporal SAEs result in a minimal loss in reconstruction metrics compared to both regular BatchTopK and Matryoshka SAEs across both models, and in some cases improve upon baselines by removing the number of dead features (Table 1). However, this slight loss in reconstruction comes at a significant improvement in the representation of semantic and contextual information.

5.3. Temporal Consistency Evaluation

We next explore the benefits of our temporal loss and find that Temporal SAEs better capture and differentiate semantic features of the data. We use high-level semantic labels from two datasets: `finefineweb` and `wikipedia`. Examples of these labels for `finefineweb` are {"literature", "food", "drama and film", "mathematics", "medical"} and examples from `wikipedia` are {"Geography", "History", "Knowledge", "People", "Religion"}. We use a part-of-speech tagger to construct syntactic labels for tokens, and also group tokens by sequence or article to understand how Temporal SAEs treat tokens from the same context.

We first conduct a qualitative evaluation by applying TSNE (Van der Maaten & Hinton, 2008) to visualize the concepts represented by Temporal SAEs and a Matryoshka SAE baseline. Language models are known to be good at tracking and recovering data over long contexts and produce semantically relevant and syntactically correct outputs, so we expect that SAEs should be able to recover all of these types of features. In Figure 1 for `finefineweb`, we see that Matryoshka SAEs only represent features corresponding to syntactic concepts, as we can only see clear clusters forming in the plot labeled with part-of-speech tags. In contrast, the plots colored by semantic category and context show no clear patterns. Temporal SAEs, on the other hand, learn a mix of all three types of features, with clusters forming in all three plots. Intuitively, clusters are most clear when labeled by context, as this is exactly what our proposed contrastive objective optimizes for. Results are qualitativaly very similar for `wikipedia`, as seen in Appendix Figure 2.

		Semantics	Syntax	Context
Pythia	Base Model	0.703	0.865	0.835
	Temporal SAE	**0.647**	0.796	**0.799**
	Matryoshka SAE	0.576	**0.848**	0.430
	BatchTopK SAE	0.587	0.847	0.654
Gemma	Base Model	0.720	0.875	0.974
	Temporal SAE	**0.682**	0.857	**0.964**
	Matryoshka SAE	0.633	**0.877**	0.895
	BatchTopK SAE	0.648	0.876	0.918

Table 2. Performance on semantic, syntactic, and context identification tasks for FineFineWeb. Temporal SAEs outperform baselines for capturing semantic and contextual information.

		Semantics	Syntax	Context
Pythia	Base Model	0.718	0.832	0.869
	Temporal SAE	**0.633**	0.770	**0.731**
	Matryoshka SAE	0.568	**0.824**	0.582
	BatchTopK SAE	0.559	0.815	0.605
Gemma	Base Model	0.939	0.873	0.927
	Temporal SAE	**0.896**	0.865	**0.882**
	Matryoshka SAE	0.882	**0.877**	0.817
	BatchTopK SAE	0.896	0.875	0.846

Table 3. Performance on semantic, syntactic, and context identification tasks for Wikipedia data. Temporal SAEs outperform baselines for capturing semantic and contextual information.

We validate these results by training probes on the SAE decompositions to predict these labels, and compare with a baseline of probing the base language model latents directly. This baseline represents how much information is natively represented in the language model; our goal is to preserve this information while transforming to a more interpretable space. Table 2 demonstrates that Temporal SAEs greatly outperform baseline methods on `finefineweb` in terms of capturing semantics and context with gains of 10% and 25% respectively for Pythia and 5% and 7% respectively for Gemma. On `wikipedia` (Table 3), we observe a similar trend. Additionally, Temporal SAEs preserve syntactical information to a similar level as the base language model. We emphasize that these SAE methods are all trained in an unsupervised manner on the same dataset, and Temporal SAEs only vary in the inclusion of a temporal consistency loss term to capture our intuition about the structure of language.

6. Conclusion

In this work, we advocate for the incorporation of priors on the data modality into the development of unsupervised methods for interpretability. We focus on the domain of natural language, leveraging the insight that semantic content remains consistent across words and sentences. We then propose a novel loss function for training SAEs such that a subset of features behave smoothly over time, better extracting semantic features from data. Through experiments, we demonstrate that the features learned by Temporal SAEs are more semantically structured, with minimal loss to reconstruction performance.

7. Acknowledgements

This work is supported in part by the National Science Foundation under grants CIF 2312667, FAI 2040880, CIF 2231707, IIS-2008461, IIS-2040989, IIS-2238714, the AI2050 Early Career Fellowship by Schmidt Sciences, and faculty research awards from Google, OpenAI, Adobe, JPMorgan, Harvard Data Science Initiative, and the Digital, Data, and Design (Dˆ3) Institute at Harvard. UB is funded by the Kempner Institute Graduate Research Fellowship. AO is supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-2140743.

References

Abdulaal, A., Fry, H., Montaña-Brown, N., Ijishakin, A., Gao, J., Hyland, S., Alexander, D. C., and Castro, D. C. An x-ray is worth 15 features: Sparse autoencoders for interpretable radiology report generation. *arXiv preprint arXiv:2410.03334*, 2024.

Alain, G. and Bengio, Y. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*, 2016.

Belinkov, Y. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219, 2022.

Biderman, S., Schoelkopf, H., Anthony, Q. G., Bradley, H., O'Brien, K., Hallahan, E., Khan, M. A., Purohit, S., Prashanth, U. S., Raff, E., et al. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pp. 2397–2430. PMLR, 2023.

Blei, D. M., Ng, A. Y., and Jordan, M. I. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan): 993–1022, 2003.

Bricken, T., Templeton, A., Batson, J., Chen, B., Jermyn, A., Conerly, T., Turner, N., Anil, C., Denison, C., Askell, A., Lasenby, R., Wu, Y., Kravec, S., Schiefer, N., Maxwell, T., Joseph, N., Hatfield-Dodds, Z., Tamkin, A., Nguyen, K., McLean, B., Burke, J. E., Hume, T., Carter, S., Henighan, T., and Olah, C. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023. https://transformer-circuits.pub/2023/monosemantic-features/index.html.

Brown, R. *A first language: The early stages*. Harvard University Press, 1973.

Bussmann, B., Leask, P., and Nanda, N. Batchtopk sparse autoencoders. *arXiv preprint arXiv:2412.06410*, 2024.

Bussmann, B., Nabeshima, N., Karvonen, A., and Nanda, N. Learning multi-level features with matryoshka sparse autoencoders. *arXiv preprint arXiv:2503.17547*, 2025.

Chanin, D., Wilken-Smith, J., Dulka, T., Bhatnagar, H., and Bloom, J. A is for absorption: Studying feature splitting and absorption in sparse autoencoders. *arXiv preprint arXiv:2409.14507*, 2024.

Dumitrescu, B. and Irofti, P. *Dictionary learning algorithms and applications*. Springer, 2018.

Elhage, N., Hume, T., Olsson, C., Schiefer, N., Henighan, T., Kravec, S., Hatfield-Dodds, Z., Lasenby, R., Drain, D., Chen, C., et al. Toy models of superposition. *arXiv preprint arXiv:2209.10652*, 2022.

Fel, T., Lubana, E. S., Prince, J. S., Kowal, M., Boutin, V., Papadimitriou, I., Wang, B., Wattenberg, M., Ba, D., and Konkle, T. Archetypal sae: Adaptive and stable dictionary learning for concept extraction in large vision models. *arXiv preprint arXiv:2502.12892*, 2025.

Gao, L., Biderman, S., Black, S., Golding, L., Hoppe, T., Foster, C., Phang, J., He, H., Thite, A., Nabeshima, N., et al. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2020.

Gao, L., la Tour, T. D., Tillman, H., Goh, G., Troll, R., Radford, A., Sutskever, I., Leike, J., and Wu, J. Scaling and evaluating sparse autoencoders. *arXiv preprint arXiv:2406.04093*, 2024.

Garcia, E. N. V. and Ansuini, A. Interpreting and steering protein language models through sparse autoencoders. *arXiv preprint arXiv:2502.09135*, 2025.

Griffiths, T., Steyvers, M., Blei, D., and Tenenbaum, J. Integrating topics and syntax. *Advances in neural information processing systems*, 17, 2004.

Hindupur, S. S. R., Lubana, E. S., Fel, T., and Ba, D. Projecting assumptions: The duality between sparse autoencoders and concept geometry. *arXiv preprint arXiv:2503.01822*, 2025.

Köhn, A. Evaluating embeddings using syntax-based classification tasks as a proxy for parser performance. In *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP*, pp. 67–71, 2016.

Makhzani, A. and Frey, B. K-sparse autoencoders. *arXiv preprint arXiv:1312.5663*, 2013.

Manning, C. and Schutze, H. *Foundations of statistical natural language processing*. MIT press, 1999.

Neville, H. J., Mills, D. L., and Lawson, D. S. Fractionating language: Different neural subsystems with different sensitive periods. *Cerebral cortex*, 2(3):244–258, 1992.

Ng, A. et al. Sparse autoencoder. *CS294A Lecture notes*, 72 (2011):1–19, 2011.

Paulo, G., Shabalin, S., and Belrose, N. Transcoders beat sparse autoencoders for interpretability. *arXiv preprint arXiv:2501.18823*, 2025.

Redington, M., Chater, N., and Finch, S. Distributional information: A powerful cue for acquiring syntactic categories. *Cognitive science*, 22(4):425–469, 1998.

Subramani, N., Suresh, N., and Peters, M. E. Extracting latent steering vectors from pretrained language models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pp. 566–581, 2022.

Team, G., Riviere, M., Pathak, S., Sessa, P. G., Hardin, C., Bhupatiraju, S., Hussenot, L., Mesnard, T., Shahriari, B., Ramé, A., et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.

Van der Maaten, L. and Hinton, G. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

Weber, M., Fu, D. Y., Anthony, Q., Oren, Y., Adams, S., Alexandrov, A., Lyu, X., Nguyen, H., Yao, X., Adams, V., Athiwaratkun, B., Chalamala, R., Chen, K., Ryabinin, M., Dao, T., Liang, P., Ré, C., Rish, I., and Zhang, C. Redpajama: an open dataset for training large language models. *NeurIPS Datasets and Benchmarks Track*, 2024.

Wu, K., Liu, J., Liu, P., and Yang, S. Time series prediction using sparse autoencoder and high-order fuzzy cognitive maps. *IEEE transactions on fuzzy systems*, 28(12):3110–3121, 2019.

A. Appendix

A.1. Additional Results.

A.1.1. Training on random previous token.

In addition to training with a constrastive loss on the previous token $\mathbf{x}_{t-1}$, we also explore training on tokens sampled uniformly at random from all past tokens $\mathbf{x}_1, ..., \mathbf{x}_{t-1}$. In Table 4 we show results on Pythia when training on a random past or the $t-1$th token. Random sampling helps recover lost performance in terms of syntax representation. We hypothesize this is because syntax is extremely predictable between two adjacent tokens $\mathbf{x}_{t-1}$ and $\mathbf{x}_t$, and random sampling breaks this predictability and requires the SAE to actually represent syntactic features.

	Semantics	Syntax	Context
Base Model	0.703	0.865	0.835
Temporal SAE (t-1)	0.647	0.796	0.799
Temporal SAE (random)	**0.687**	0.839	**0.830**
Matryoshka SAE	0.576	**0.848**	0.430
BatchTopK SAE	0.587	0.847	0.654

Table 4. Table 2 from main paper with additional results for Temporal SAE trained on randomly-sampled past token.

A.1.2. TSNE plots for wikipedia.

In Figure 2, we report TSNE plots on the `wikipedia` dataset. Similar to `finefineweb`, we observe that Temporal SAEs contain more semantic structure than baselines.

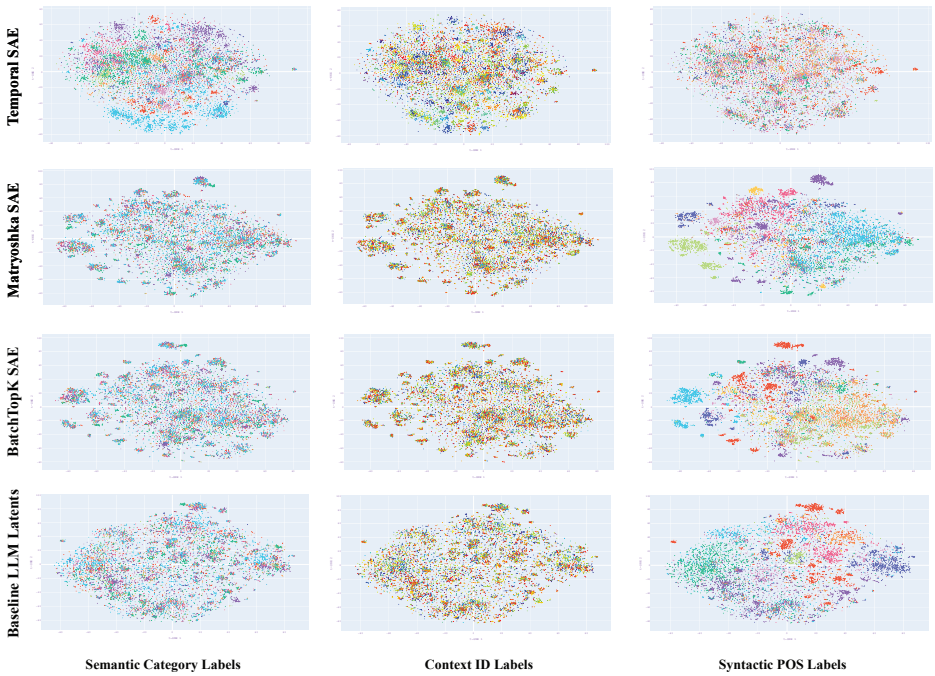

Figure 2. TSNE plots of the SAE feature activation for all SAE architectures and the LLM Latent Baseline (bottom). We label points by their semantic category (left), the context or sequence the token comes from (middle), and the part of speech of the token (right). We find that Temporal SAEs learn a balance of both semantic and syntactic information, whereas other SAEs prioritize only syntactic information at the cost of all semantics.

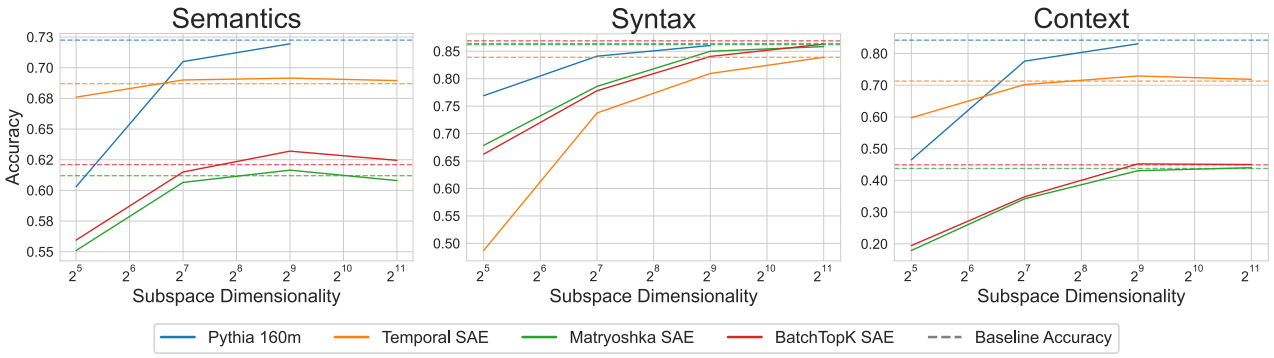

Figure 3. Probe performance on Semantics, Syntax, and Context after projecting via PCA to varying subspace dimensionalities. Note that Pythia's latent space has a maximum projected of $2^9 = 512$ as its original dimensionality is 768.

A.1.3. Probing with varying feature subspace sizes.

We explore how semantic, syntactic, and contexual information are distributed in the representations of Pythia and its SAEs. An interpretable monosemantic representation should use a few (and ideally one) neuron to represent a feature, and we test this by projecting the SAE representations into a lower-dimensional subspace and measure the drop in performance of probing. Additionally, we apply the same projection analysis to the latent representation of the base model, and under the hypothesis that the language model neurons are highly polysemantic and represent features in superposition, the performance drop of the base model should be relatively larger than that of an SAE.

We project into the optimal lower-dimensional subspace that contributes most to the variance of our data distribution using PCA. Specifically, for each dataset, we compute a PCA of the data matrix and then retain only the top k components for varying amounts of k. We then train probes with ℓ_2 regularization on the base dataset and the filtered dataset for each level of k and report test accuracy (on an 80%-20% train-test split) in Figure 3. We observe that the probe accuracy on the semantic task increases more rapidly for the base model than for any of the SAEs, validating the superposition hypothesis: signal for the semantic task is distributed across more features, so the baseline model needs a relatively higher set of principal components to recover its full-latent performance. We observe a similar trend for the context task. Finally, for the syntax task, we observe that Temporal SAE performance is close-behind the base model and baseline SAEs, although it also has a sharp increase in accuracy (indicating a more distributed syntax representation) than baselines.