Decoding Dark Matter: Specialized Sparse Autoencoders for Interpreting Rare Concepts in LLMs

Aashiq Muhamed
Language Technologies Institute
Carnegie Mellon University
amuhamed@andrew.cmu.edu

Jake Mendel
Apollo Research
jake@apolloresearch.ai

Lucius Bushnaq
Apollo Research
lucius@bushnaq.de

Mona Diab
Language Technologies Institute
Carnegie Mellon University
mdiab@andrew.cmu.edu

Virginia Smith
Machine Learning Department
Carnegie Mellon University
smithv@andrew.cmu.edu

Abstract

Understanding and mitigating the potential risks associated with large language models (LLMs) hinges on the development of effective interpretability methods. Sparse Autoencoders (SAEs) have emerged as a promising tool for disentangling LLM representations, but they often struggle to capture rare, yet crucial, features, especially those relevant to safety. We introduce Specialized Sparse Autoencoders (SSAEs), a novel approach designed to illuminate these elusive "dark matter" features by focusing on specific subdomains. We present a practical recipe for training SSAEs, demonstrating the efficacy of Dense retrieval for data selection and the benefits of Tilted Empirical Risk Minimization (TERM) as a training objective. We evaluate SSAEs on standard metrics, such as downstream perplexity and L0 sparsity, and find that they effectively capture subdomain tail concepts, exceeding the capabilities of general-purpose SAEs. Furthermore, TERM-trained SSAEs yield more interpretable features, as evidenced by our automated evaluation using LLMs to generate and assess feature explanations. SSAEs, particularly those trained with TERM, provide a powerful new lens for peering into the inner workings of LLMs in subdomains and hold significant promise for enhancing AI safety research.

1 Introduction

Interpretability is crucial for ensuring the safety and reliability of large language models (LLMs). Sparse Autoencoders (SAEs) have emerged as a promising tool for disentangling the complex, high-dimensional representations within LLMs into meaningful, interpretable features [9, 13, 7, 8]. However, recent work [29] suggests that even massively wide SAEs, trained on vast amounts of data, may only be scratching the surface in terms of capturing the full spectrum of concepts present in these model representations. A significant portion of rare or highly specific concepts remain essentially invisible due to their infrequent activation. These elusive features, akin to "dark matter" in the universe of interpretability, pose a significant challenge for understanding and mitigating potential risks associated with LLMs. While larger SAEs did exhibit some features for rarer concepts, the researchers found compelling evidence suggesting a vast amount of "dark matter" features were still being missed. For example, they found features for some of San Francisco's neighborhoods, but their model still lacked features for smaller entities like coffee shops or street intersections. They observed that if a concept is present only once every billion tokens, we may need a billion-feature SAE to capture it reliably. This raises a critical question: can we develop more efficient methods than simply scaling SAE width to capture the tail concepts we are interested in, particularly those relevant to safety?

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

This paper introduces Specialized Sparse Autoencoders (SSAEs), a novel approach designed to address this challenge. Instead of aiming to capture all concepts, as in current SAE practices, we propose SSAEs as a more targeted method for efficiently extracting rare features related to specific subdomains, especially those crucial for safety analyses. We show that by focusing on a particular subdomain, we can train SSAEs to learn features representing tail concepts without needing to scale to billions of features. Furthermore, instead of relying solely on scaling, we investigate whether Tilted Empirical Risk Minimization (TERM), which approximates minimax risk at large tilt parameters, can further improve the representation of tail concepts within SSAEs. Our primary contributions are threefold:

1. **Specialized Sparse Autoencoders:** We introduce SSAEs, a new approach for training SAEs specifically designed to capture tail concepts within a given subdomain. We demonstrate empirically that SSAEs capture a greater proportion of tail concepts compared to standard SAEs trained on general-purpose data.
2. **Subdomain Data Selection Strategies:** We present a practical recipe for training SSAEs, starting with a small seed dataset and leveraging various data selection strategies to identify relevant training data from the LLM's pretraining corpus. Our investigation reveals that Dense retrieval is a particularly effective strategy, while TracIn reranking can offer further improvements.
3. **Tilted Empirical Risk Minimization for SAEs:** We propose Tilted Empirical Risk Minimization (TERM) as a novel training objective for SAEs. At large tilt values, TERM encourages a more balanced learning of both head and tail concepts. We show that TERM leads to more interpretable features, while maintaining comparable downstream perplexity performance to ERM-trained SAEs.

We envision SSAEs serving as versatile concept detectors within various AI safety applications. They can be used to identify and analyze rare, potentially dangerous concepts, such as those related to deception or harmful biases. Additionally, SSAEs could be employed to unlearn specific concepts from a model or integrated with other SAEs in a mixture-of-experts fashion to achieve more comprehensive interpretability.

2 Methodology

2.1 Sparse Autoencoders (SAE)

The superposition hypothesis in LLMs suggests that a limited number of neurons encode a much larger number of concepts, leading to complex and overlapping representations [12]. Superposition, while efficient, makes it challenging to interpret individual neuron representations or directions in representation space. Sparse autoencoders (SAEs) offer a potential solution by learning to reconstruct LLM representations at a layer using a sparse set of features in a higher-dimensional space, potentially disentangling superposed features and revealing more interpretable representations [11, 22]. In a well-trained SAE, individual features in the hidden dimension align with underlying sparse, semantically meaningful features [10].

SAEs decompose a model's activation $x \in \mathbb{R}^n$ into a sparse, linear combination of feature directions: $x \approx x_0 + \sum_{i=1}^{M} f_i(x) d_i$, where d_i are $M \gg n$ latent unit-norm feature directions, and the sparse coefficients $f_i(x) \geq 0$ are the corresponding feature activations for x. The right-hand side of this equation has the structure of an autoencoder: an input activation x is encoded into a (sparse) feature activations vector $f(x) \in \mathbb{R}^M$, which is then linearly decoded to reconstruct x. We parameterize a single-layer autoencoder $(f, \hat{x})$ as follows: $f(x) := \text{ReLU}(W_{enc}(x) + b_{enc})$ and $\hat{x}(f) := W_{dec} f + b_{dec}$ where $W_{enc} \in \mathbb{R}^{M \times n}$ and $W_{dec} \in \mathbb{R}^{n \times M}$ are the encoding and decoding weight matrices, and $b_{enc} \in \mathbb{R}^M$ and $b_{dec} \in \mathbb{R}^n$ are the bias vectors. The training objective combines a reconstruction loss and a sparsity penalty:

$$L(x) := \|x - \hat{x}(f(x))\|_2^2 + \lambda \|f(x)\|_1 \tag{1}$$

where $\lambda > 0$ is a hyperparameter controlling the trade-off between reconstruction fidelity and sparsity. We constrain the columns of W_{dec} to have unit norm during training [8].
In existing work, SAEs for LLMs are trained on the same large, general-purpose dataset used to train the underlying language model [8, 9, 25, 13]. This approach ensures that the SAE captures a wide array of concepts present in the general language domain. However, this can result in the SAE learning features that are frequent in the pretraining data but miss concepts within specific domains of interest, especially those that are rare by frequency in the pretraining data.

2.2 Specialized Sparse Autoencoders (SSAE)

This work introduces Specialized Sparse Autoencoders (SSAEs), designed to learn features representing rare concepts within specific subdomains. Our approach begins with a small seed concept dataset, comprising either a specific concept or limited data from the target subdomain (e.g., toxicity). We then expand this seed dataset using a high-recall retrieval strategy that leverages the seed data to identify and retrieve relevant examples from the LLM's pretraining corpus. To create an SSAE, we finetune a pretrained general-purpose SAE (GSAE) on this curated subdomain data using Equation 1. The GSAE is initially trained to reconstruct activations on a large, general-purpose dataset, enabling it to capture a broad range of concepts. Finetuning on the subdomain data allows the SAE to specialize and learn features that may be infrequent in the general domain but prevalent within the target subdomain.

To evaluate the quality of the trained SAEs, we use two metrics: L_0 and Perplexity with SAE [8]. L_0 measures the sparsity of the SAE and is defined as the average number of active features on a given input, i.e. $\mathbb{E}_{x\sim D}\|f(x)\|_0$. Perplexity with SAE measures the reconstruction fidelity of the SAE and is the average cross-entropy loss of the language model on an evaluation dataset, when the SAE's reconstructions are spliced into it. A better SAE recovers more of the base model's performance. All other things being equal, a better SAE needs fewer features (L_0) to explain model performance on a given datapoint. Unlike existing works that evaluate SAEs on subsampled training data, we evaluate SSAE generalization using both in-distribution and out-of-distribution test sets drawn from the same subdomain. This dual evaluation approach assesses the SSAE's ability to both accurately capture concepts within the specific training data distribution and generalize to unseen data, reflecting the capability to learn broader subdomain concepts.
Specialization allows SSAEs to effectively uncover and interpret rare concepts that might be overlooked by traditional SAEs trained solely on general-purpose data. Consequently, SSAEs provide a valuable tool for detecting, understanding, and potentially mitigating risks associated with the emergence of rare concepts within LLM representations.

2.3 Subdomain Data Selection Strategies

The effectiveness of SSAEs depends crucially on the quality and relevance of the selected subdomain data used for finetuning. We study several data selection strategies, leveraging both sparse and dense retrieval methods to identify data points from a larger corpus (the LLM's pretraining data) that are most relevant to the seed data:

Sparse Retrieval: Okapi BM25 [26], an advanced TF-IDF variant, ranks documents based on relevance to a query, considering term frequency, inverse document frequency, and document length. We use the seed dataset as a query to retrieve relevant documents from the larger corpus.

Dense Retrieval: Contriever [16], a dual-encoder based dense retriever, generates semantically meaningful embeddings for queries and documents. We embed the seed dataset and candidate documents, using cosine similarity to retrieve documents most similar to the seed concepts.

SAE TracIn: Training data Influence Score (TracIn) [24] quantifies the influence of training examples on model predictions. We adapt TracIn to SAEs by calculating the dot product of the loss gradients with respect to the training data and seed data: $\mathrm{TracIn}\left(z, z'\right) = \nabla L_w\left(z\right) \cdot \nabla L_w\left(z'\right)$ where z is a training data point, z' is the seed dataset, w are the pretrained SAE weights, and $L_w(\cdot)$ is the SAE loss (Equation 1). We identify influential data points from the larger corpus using a two-stage approach: Initial Filtering with Sparse or Dense retrieval, followed by TracIn Reranking to select the most influential data points for training the SSAE.

2.4 Refinement with Tilted Empirical Risk Minimization

Standard Empirical Risk Minimization (ERM) during finetuning tends to prioritize learning features for the most frequent ("head") concepts in the subdomain data. However, for safety applications, capturing rare ("tail") concepts is often crucial. To address this, we utilize Tilted Empirical Risk Minimization (TERM) [21, 3]. TERM modifies the standard ERM objective by introducing a tilt parameter (t) that controls the emphasis on different parts of the loss distribution: $\tilde{L}(t; w) = \frac{1}{t}\log\left(\frac{1}{N}\sum_{i\in[N]} e^{t\cdot L_w(z_i)}\right)$ where $L_w(z_i)$ is the standard SAE loss for data point z_i and SAE parameters w.

TERM generalizes ERM as the 0-tilted loss recovers the average loss, while it also recovers other alternatives such as the max-loss ($t \to +\infty$) and min-loss ($t \to -\infty$). In this work we use large tilt parameters ($t \gg 0$) to effectively minimize the maximum loss, encouraging the model to learn features that better represent the tail of the data distribution, including rare concepts. Incorporating TERM during finetuning leads to a more balanced representation of both head and tail concepts within the subdomain. We find that TERM can also lead to more compositional and interpretable features. This is because, instead of creating many specific features for slightly different variations of tail concepts, TERM can encourage the SAE to learn a smaller number of more general features that capture the essence of these concepts.

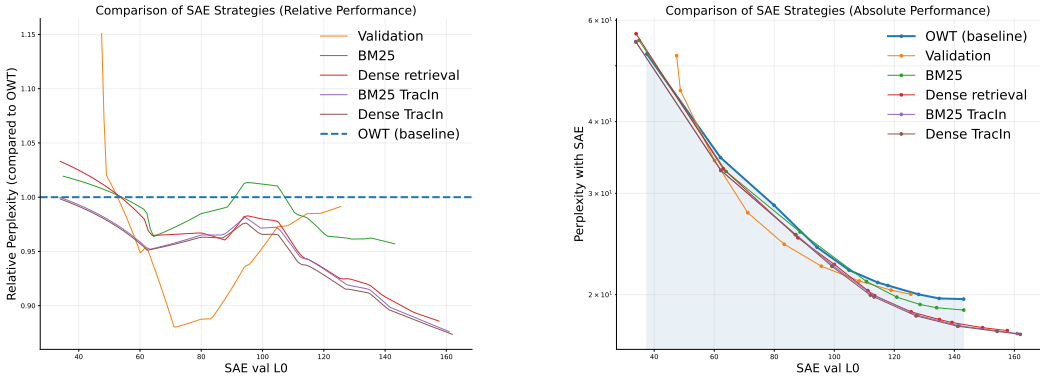
3 Experiments And Results

3.1 Experimental Setup

We conduct our experiments using the publicly available pretrained Gemma-2b [28] residual stream GSAE, specifically the 'gemma-2b-res-jb' checkpoint and the 'blocks.12.hook_resid_post' layer [5]. These SAEs have a feature width of 16384 and were pretrained on OpenWebText (OWT) [14]. For plotting the Pareto front, we sweep over a range of 8 L1 penalty coefficients and choose the best-performing model based on the validation split for each L1 value. The selected models are then evaluated on the held-out test split.
All SAEs are trained using the Adam optimizer [19] with a learning rate of 5e-5. We employ a token batch size of 4096 and shuffle the data within a batch buffer of size 4. We use a linear learning rate decay schedule over the last 1000 steps. All experiments can be completed in under 12 hours using four A6000 GPUs. We utilize the SAELens [6] library for our SAE training and analysis.

3.2 Data Selection Strategies

We evaluate the effectiveness of various data selection strategies for training SSAEs.



(a) (b)

Figure 1: Pareto curves for Physics SSAE trained with various data selection strategies as the sparsity coefficient is varied on arXiv Physics test data. We plot (a) Perplexity with spliced in SAE relative to GSAE (baseline) (b) Absolute Perplexity with the spliced in SSAE. Dense TracIn and BM25 TracIn achieve comparable performance, performing slightly Dense retrieval, which in turn outperforms BM25 retrieval and OWT Baseline. All curves are averages over three SAE training seeds.

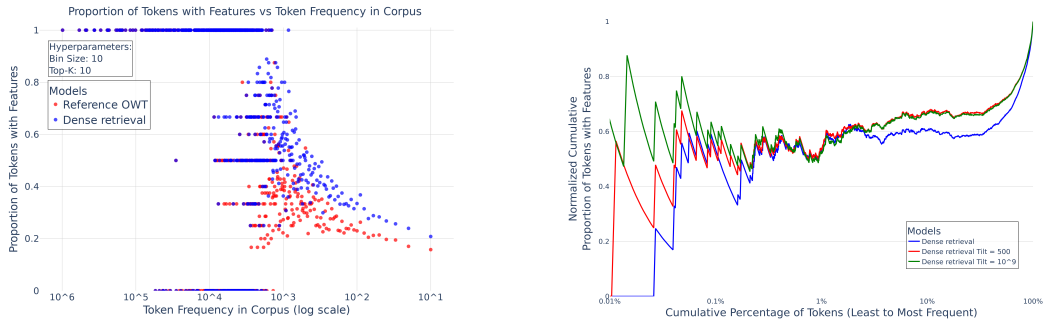
SSAE for Physics We start with a seed concept dataset (Validation) consisting of 9.2K tokens sampled from the arXiv Physics dataset [1]. We then employ three data selection strategies–Sparse Retrieval, Dense Retrieval, and SAE TracIn to expand this seed dataset and retrieve 13.9M tokens from the OWT. The SSAE is trained on this expended set by finetuning the GSAE for 1000 iterations. For SAE TracIn, we first reduce the candidate pool to 1% of OWT using either BM25 or Dense retrieval. Then, we rerank this filtered set using TracIn scores and select the most influential data points for training the SSAE, referring to these methods as BM25 TracIn and Dense TracIn, respectively.

We train an SSAE for each strategy and compare its performance to a baseline SAE finetuned on the full OWT dataset (baseline) across various sparsity coefficients (λ). We evaluate the models on two test splits: 4.8M tokens from the arXiv Physics dataset (in-distribution) and 700K tokens from the Physics instruction tuning dataset [15](out-of-distribution). Testing on instruction data helps measure whether the SAEs are overfitting to the specific template of the text as opposed to identifying concepts. Figure 1 and 4 show the patched perplexity vs. L_0 curves for these experiments.

We measure performance using area under the curve for a range of L0 from 60 to 140 i.e., a selection strategy with lower perplexity when the SSAE is spliced in is better. Our findings indicate that Dense TracIn and BM25 TracIn achieve comparable performance, surpassing Dense retrieval alone, which in turn outperforms BM25 retrieval. Training on the full OWT dataset yields the lowest performance. We observe: (a) Dense retrieval outperforms BM25. SSAEs trained on data selected with Dense Retriever consistently achieve lower perplexity for a given L_0 than those trained with BM25, both in and out of distribution. (b) BM25 exhibits poor out-of-distribution generalization. While BM25 performs reasonably well on the in-distribution test set, its performance degrades significantly on the out-of-distribution test set. (c) Multiple passes on seed data (Validation) during SSAE training improves in-distribution performance but degrade out-of-distribution performance. This suggests multiple passes can overfit to the structure or template of the seed dataset. (d) While TracIn reranking after Dense retrieval yields a marginal performance gain, Dense retrieval alone proves to be highly competitive.

SSAE for Toxicity We repeat the experiment using a seed concept dataset of 4072 tokens from the Pile Toxicity dataset [20]. We retrieve 5.25M tokens from OWT using the same strategies as before and train SSAEs on this data for 500 iterations. We then evaluate the models on a test split of 3.357M tokens from the Pile Toxicity dataset (in-distribution).
Appendix Figure 5 displays the patched perplexity versus L_0 curves for these experiments. The results largely align with the physics experiment, with Dense retrieval outperforming BM25 and TracIn offering a marginal improvement over Dense retrieval alone.

3.3 Specialized SAEs and Tail Concept Learning

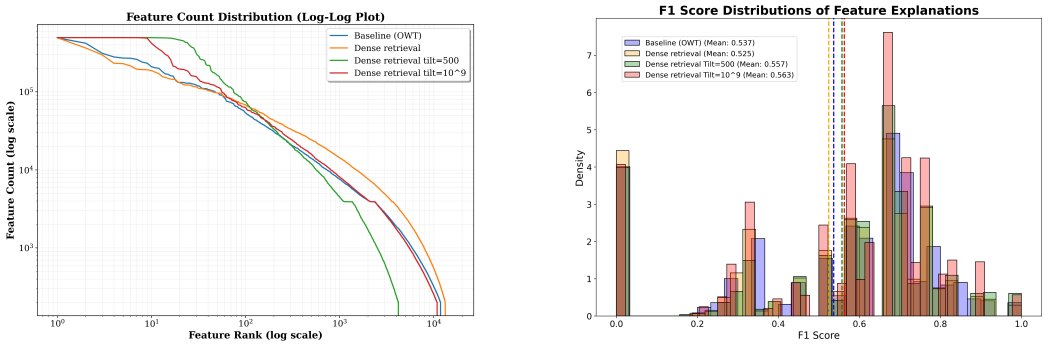


(a) (b)

Figure 2: (a) Proportion of tokens with SAE features vs. Token frequency in Physics arXiv data. SSAE trained with dense retrieval captures more tail tokens (concepts) in its features. (b) Cumulative proportion of tokens with SAE features vs. cumulative percentage of tokens in Physics arXiv data, normalized per model so that the cumulative proportion of tokens with features is 1 over the entire dataset. SSAE trained with dense retrieval and larger tilt captures more tail tokens (concepts) in its features.

In Figure 2(a), we leverage the unembedding matrix as a logit lens to analyze the top-10 token logits associated with each SSAE feature [17]. For each frequency bucket in the Physics arXiv test data, we calculate the percentage of tokens that appear among the top-10 logits for at least one feature. This analysis allows us to assess the extent to which SSAE features represent tokens across different frequency ranges. We compare two SSAEs at the same test L_0 of 100: one finetuned on the full OWT dataset and another finetuned using Dense retrieval. Our findings reveal that the Dense retrieval finetuned SSAE captures a significantly higher proportion of tail tokens in its features compared to the OWT finetuned SSAE. Moreover, these captured tail tokens often correspond to physics-specific concepts, suggesting that SSAEs are indeed learning to represent rare, domain-relevant concepts.

3.4 Can Tilted ERM further learn tail concepts?



(a) (b)

Figure 3: (a) **Feature activation count vs. feature rank** for SSAEs trained on the Physics arXiv dataset using different strategies: full OWT, Dense retrieval, and Dense retrieval with tilt. Tilt encourages the learning of more broadly activating features, indicating increased concept coverage and recall. (b) **Automated interpretability**: F1 score distributions for feature activation prediction on the Physics arXiv dataset, based solely on LLM-generated feature explanations. An LM is provided with examples that activate a feature and is asked to generate an explanation. These explanations are then used to predict feature activations on new examples. Dense retrieval with tilt yields explanations that are more predictive compared to the OWT baseline and Dense retrieval alone.

While standard ERM finetuning of SSAEs on Dense retrieval data improves tail concept coverage compared to GSAEs, it still prioritizes learning head concepts. To address this and further enhance tail concept representation, we investigate TERM. At high tilt parameters, TERM minimizes maximum risk, encouraging the model to learn features that better capture the tail of the data distribution.

Figure 2(b) plots the cumulative proportion of tokens with SAE features (identified using the logit lens approach [17]) versus the cumulative percentage of tokens in the Physics arXiv data. We normalize the curves per model at a validation L_0 of 100, ensuring that the cumulative proportion of tokens with features reaches 1 over the entire dataset. Results show that SSAEs trained with Dense retrieval and tilt capture a greater proportion of tail tokens compared to Dense retrieval alone, with the effect increasing with tilt. Figure 7 presents the histogram of differences in feature activation counts for the same features between SSAEs and the OWT baseline. Comparing SSAEs trained with Dense retrieval and Dense retrieval with tilt, we observe that the tilted SSAE exhibits a greater shift towards higher activation counts, indicating more pronounced learning of domain-specific features. Figure 3(a) further analyzes feature activation by plotting feature activation count versus feature rank, demonstrating that TERM encourages learning of more broadly activating features, suggesting increased concept coverage and recall. This represents a fundamentally different mechanism for feature learning compared to standard ERM, promoting more compositional features that improve recall and capture tail features.

Capturing rare concepts is not synonymous with having rare features (i.e., features that only activate on a few data points). Consider training an SAE with a single feature capacity on data representing online gaming communication. Let's define three concepts: "general trash talk" (frequent), "targeted harassment" (less frequent), and "credible threats of violence" (rare). A GSAE would likely learn a feature that fires predominantly for "trash talk" due to its prevalence, neglecting the more serious "targeted harassment" and "credible threats". An SSAE trained with standard ERM might exhibit similar behavior. However, a tilted SSAE, by effectively minimizing the maximum risk, could learn a single, more general feature representing "toxic behavior" that represents all three concepts more equally. Tilted SSAEs can therefore improve concept coverage by mitigating the dominance of frequent concepts and ensuring a more balanced representation that includes both common and rare, but equally harmful, behaviors within a unified feature.

Figure 6 shows that TERM-finetuned SSAEs achieve comparable downstream perplexity to ERM-trained SSAEs within a specific L_0 regime (85-100). However, outside this range, our current training methodology faces challenges in precisely controlling L_0 using only the sparsity penalty, potentially resulting in decreased performance or numerous inactive features. The effectiveness of tilted ERM can be attributed to its connection to minimax losses, which improve robustness

and out-of-distribution (OOD) generalization [30, 27]. By optimizing for the worst-case scenario, minimax methods encourage learning of more invariant features that generalize better to unseen data. We argue that interpretability, particularly for detecting rare safety-relevant features, is fundamentally an OOD problem, as these features are often underrepresented in the training data.

3.5 Automated Interpretability

SAEs offer a framework for more interpretable representations, leading researchers to explore methods for automatically interpreting the features they learn. Existing works [4, 29] have employed LLMs like GPT-4 to generate neuron explanations based on strongly activating text, or analyze SAE features.

We employ a sequence-level classification task for interpretability evaluation [18]. Instead of predicting feature activation at each token, we task an LLM with identifying whether entire sequences contain a given feature. This simplified task requires fewer few-shot examples, fewer input/output tokens, and allows for the use of smaller, faster LLMs while maintaining reliable scores. We utilize Claude 3.5 Sonnet [2] as both the *Interpreter* and the *Predictor* in our automated interpretability framework. The Interpreter generates explanations for each feature based on the top 10 activating examples (see Appendix E for examples). Subsequently, the Predictor receives these explanations along with 5 examples drawn from the quintiles of the top activating examples (not used for explanation generation) and 5 randomly selected non-activating examples. The Predictor is then tasked with predicting whether each example activates the feature (see Appendix F for the LLM prompts). We evaluate the interpretability of the explanations by measuring the F1 score between the Predictor's predictions and the true feature activations.

Figure 3(b) presents F1 score distributions for feature activation prediction on the Physics arXiv dataset, based solely on LLM-generated explanations. Dense retrieval with tilt consistently achieves higher F1 scores compared to both the OWT baseline and Dense retrieval alone, indicating that explanations generated for these features are more effective in predicting activation on new examples. Interestingly, while SSAEs with dense features demonstrated superior performance in terms of downstream perplexity versus L_0, their feature explanations were not necessarily more interpretable. This observation aligns with findings in existing work [23], that observed a decrease in interpretability (measured by Pearson correlation) with increasing SAE width, attributed to the learning of fine-grained features that are challenging to interpret. Since TERM tends to encourage the learning of coarser and more compositional features, we find that the resulting explanations tend to be more readily interpretable.

4 Conclusion and Future Work

This work introduces SSAEs, a novel approach for interpreting rare, subdomain-specific features in LLMs. SSAEs trained with Dense retrieval and TERM outperform standard SAEs in capturing tail concepts, while also yielding more interpretable features. We believe SSAEs hold significant potential for advancing AI safety by enabling the detection and analysis of potentially harmful or unexpected LLM behaviors. Future work could investigate leveraging SSAEs for targeted concept unlearning.

5 Acknowledgements

This research was supported by the Anthropic Researcher Access Program through their generous grant of model credits, and AI Safety Support. The project originated during Aashiq's participation in the ML Alignment and Theory Scholars (MATS) program.

References

[1] Anonymous. arxiv physics dataset. `https://huggingface.co/datasets/anonymousdatasets/arxiv-physics`, 2024.

[2] Anthropic. Claude 3.5 sonnet. `https://www.anthropic.com` or `https://claude.ai`, 2024. AI model.

[3] Ahmad Beirami, Robert Calderbank, Mark M Christiansen, Ken R Duffy, and Muriel Médard. A characterization of guesswork on swiftly tilting curves. *IEEE Transactions on Information Theory*, 65(5):2850–2871, 2018.

[4] Steven Bills, Nick Cammarata, Dan Mossing, Henk Tillman, Leo Gao, Gabriel Goh, Ilya Sutskever, Jan Leike, Jeff Wu, and William Saunders. Language models can explain neurons in language models. Technical report, 2023. Accessed: 14.05.2023.

[5] John Bloom. Gemma-2b-residual-stream-saes. `https://huggingface.co/jbloom/Gemma-2b-Residual-Stream-SAEs`, 2024.

[6] Joseph Bloom and David Chanin. Saelens. `https://github.com/jbloomAus/SAELens`, 2024.

[7] Dan Braun, Jordan Taylor, Nicholas Goldowsky-Dill, and Lee Sharkey. Identifying functionally important features with end-to-end sparse dictionary learning. *arXiv preprint arXiv:2405.12241*, 2024.

[8] 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.

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

[10] David L Donoho. Compressed sensing. *IEEE Transactions on information theory*, 52(4):1289–1306, 2006.

[11] Nelson Elhage, Tristan Hume, Catherine Olsson, Neel Nanda, Tom Henighan, Scott Johnston, Sheer ElShowk, Nicholas Joseph, Nova DasSarma, Ben Mann, Danny Hernandez, Amanda Askell, Kamal Ndousse, Andy Jones, Dawn Drain, Anna Chen, Yuntao Bai, Deep Ganguli, Liane Lovitt, Zac Hatfield-Dodds, Jackson Kernion, Tom Conerly, Shauna Kravec, Stanislav Fort, Saurav Kadavath, Josh Jacobson, Eli Tran-Johnson, Jared Kaplan, Jack Clark, Tom Brown, Sam McCandlish, Dario Amodei, and Christopher Olah. Softmax linear units. *Transformer Circuits Thread*, 2022. https://transformer-circuits.pub/2022/solu/index.html.

[12] Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, et al. Toy models of superposition. *arXiv preprint arXiv:2209.10652*, 2022.

[13] 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.

[14] Aaron Gokaslan, Vanya Cohen, Ellie Pavlick, and Stefanie Tellex. Openwebtext corpus. `http://Skylion007.github.io/OpenWebTextCorpus`, 2019. Accessed: September 17, 2024.

[15] Algorithmic Research Group. arxiv physics instruct tune 30k dataset. `https://huggingface.co/datasets/AlgorithmicResearchGroup/arxiv-physics-instruct-tune-30k`, 2024.

[16] Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. Unsupervised dense information retrieval with contrastive learning. In *Transactions of the Association for Computational Linguistics*, volume 10, pages 726–741. MIT Press, 2022.

[17] Johnny Lin Joseph Bloom. Understanding sae features with the logit lens. `https://www.lesswrong.com/posts/qykrYY6rXXM7EEs8Q/understanding-sae-features-with-the-logit-lens`, 2024.

[18] Caden Juang, Gonçalo Paulo, Jacob Drori, and Nora Belrose. Open source automated interpretability for sparse autoencoder features. `https://blog.eleuther.ai/autointerp/`, July 2024. EleutherAI Blog.

[19] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

[20] Tomek Korbak. Pile toxicity balanced dataset. `https://huggingface.co/datasets/tomekkorbak/pile-toxicity-balanced`, 2024.

[21] Tian Li, Ahmad Beirami, Maziar Sanjabi, and Virginia Smith. Tilted empirical risk minimization. *arXiv preprint arXiv:2007.01162*, 2020.

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

[23] Charles O'Neill, Christine Ye, Kartheik Iyer, and John F Wu. Disentangling dense embeddings with sparse autoencoders. *arXiv preprint arXiv:2408.00657*, 2024.

[24] Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. Estimating training data influence by tracing gradient descent. In *Advances in Neural Information Processing Systems*, volume 33, pages 19920–19930, 2020.

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

[26] Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389, 2009.

[27] Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks for group shifts: On the importance of regularization for worst-case generalization. *arXiv preprint arXiv:1911.08731*, 2019.

[28] Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, et al. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*, 2024.

[29] 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, Alex Tamkin, Esin Durmus, Tristan Hume, Francesco Mosconi, 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. Technical report, Anthropic, 2024.

[30] Haotian Ye, Chuanlong Xie, Tianle Cai, Ruichen Li, Zhenguo Li, and Liwei Wang. Towards a theoretical framework of out-of-distribution generalization. *Advances in Neural Information Processing Systems*, 34:23519–23531, 2021.

A Evaluating Specialized SAE for Physics on OOD data

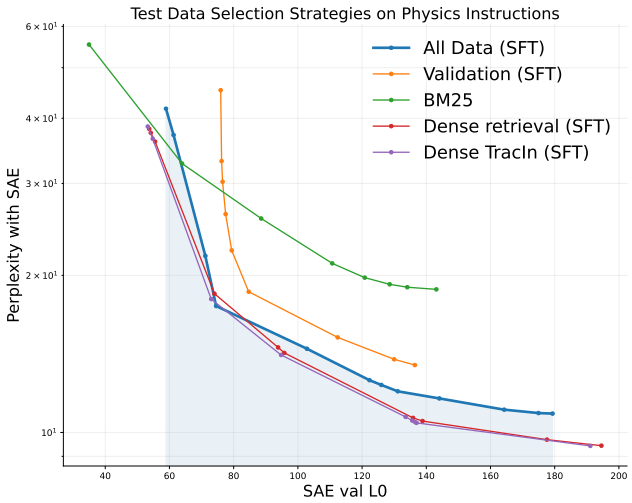
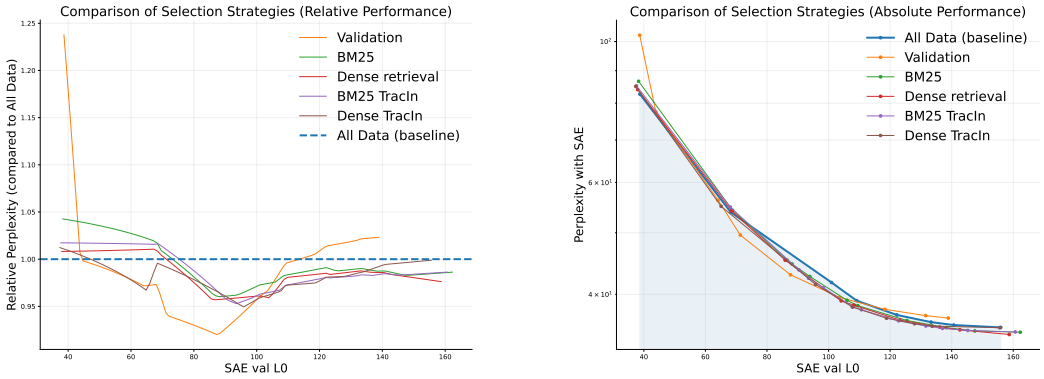


Figure 4: Pareto curves for SSAE trained with various data selection strategies as the sparsity coefficient is varied on Physics instruction test data. We plot absolute perplexity with the spliced in SSAE. We find that both BM25 retrieval and training on the validation data generalize poorly when tested out of domain.

B Specialized SAE for Toxicity



(a) (b)

Figure 5: Pareto curves for Toxicity SSAE trained with various data selection strategies as the sparsity coefficient is varied on Pile toxicity test data. We plot (a) Perplexity with spliced in SAE relative to a GSAE (Baseline) (b) Absolute Perplexity with the spliced in SSAE. Dense TracIn achieves the best performance, followed by Dense retrieval, BM25 TracIn, BM25 and OWT baseline.

C Evaluating Tilted ERM SAE

Figure 6 evaluates SSAEs trained with Tilted ERM, displaying Pareto curves where the x-axis represents L0 and the y-axis shows downstream perplexity with patched-in SAE.

D Relative Feature Activation Distribution

Figure 7 plots the histogram of differences in feature activation counts between specialized SSAEs and the OWT baseline on the Physics arXiv dataset.

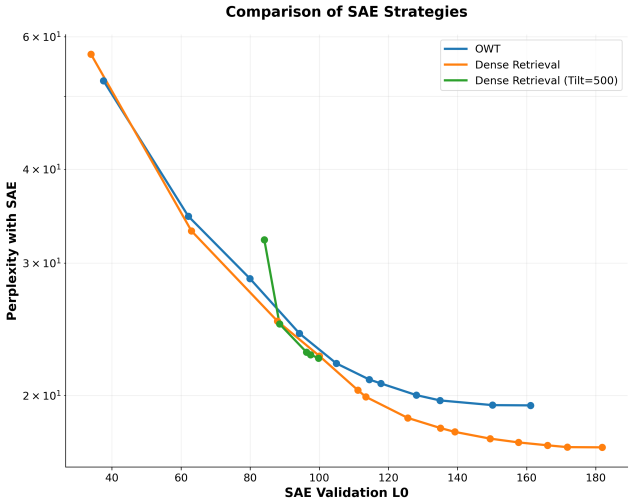


Figure 6: Pareto curves for SSAEs finetuned on the Physics arXiv dataset using different strategies: full OpenWebText (OWT), Dense retrieval, and Dense retrieval with Tilted Empirical Risk Minimization (TERM, tilt=500). TERM-finetuned SSAEs achieve competitive performance with Dense retrieval alone within the L_0 range of 85-100. Outside this range, our current training methodology results in a high percentage of inactive (dead) features.

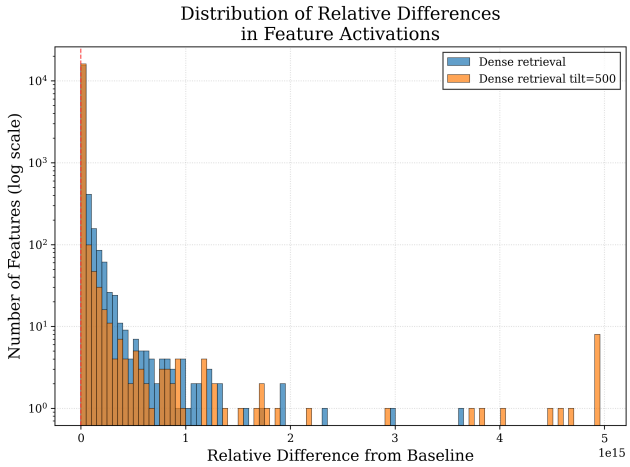


Figure 7: Histogram of differences in feature activation counts for the same features between specialized SSAEs and the OWT baseline on the Physics arXiv dataset. We compare SSAEs trained with Dense retrieval (blue) and Dense retrieval with tilt (orange). Positive values indicate features activating on more data points in the specialized models compared to the baseline, suggesting adaptation to the physics domain. The tilted SSAE exhibits a greater shift towards higher activation counts, indicating a more pronounced learning of domain-specific features.

E Automated Intepretability Explanations

We list the feature explanations generated by the Interpreter for the first ten features of the GSAE and SSAE, considering only those that were activated on the arXiv Physics test set. We observe that the SSAE specializes its features to handle a wider range of cases, but these specialized feature explanations are more complex and less easily understood. We find that the tilted SSAEs produce feature explanations that are easier to interpret, although they have fewer active features on the test set. This may be because the tilted SSAE learns more compositional features, some of which may be rare.

Generalized SAE

0. The token "0" appearing in scientific notation, journal article citations, or encoded ASCII representations, often in the context of physics or chemistry literature references.

5. This neuron appears to activate on mathematical and scientific notation, particularly symbols, equations, and specialized formatting in technical documents. It may play a role in recognizing and processing scientific or mathematical content within text.

7. The neuron appears to activate on punctuation marks, particularly commas and quotation marks, when they are used to separate or enclose items in a list, mathematical expressions, or technical notation in scientific or mathematical text. It may play a role in parsing and understanding the structure of complex technical writing.

8. This neuron appears to activate on tokens that are part of or follow noun phrases, often in technical or academic contexts. It seems to be sensitive to words that introduce or refer to specific objects, concepts, or pieces of information within a larger text. The neuron may play a role in tracking referential elements or key pieces of information in complex, information-dense text.

9. The token "," appearing after complex scientific or technical phrases, often preceding conjunctions or additional clauses that provide further explanation or context in academic or scientific writing.

10. This neuron appears to activate on abbreviated references to academic or scientific sources, particularly in bibliographies or citation lists. It responds to: 1. Abbreviated journal names (e.g. "NY", "APS", "Euro") 2. Abbreviated organization names (e.g. "SIAM", "INSPEC") 3. URL components of online references (e.g. "citeseer", "philsci", "biology-") 4. Abbreviated publisher names (e.g. "TERRAPUB") The neuron seems to play a role in recognizing citation patterns.

Specialized SAE

0. The token "0" appearing in scientific paper citations, journal volume numbers, or ASCII code representations, often in the context of physics or mathematics literature.

4. This neuron appears to activate on tokens related to academic and scientific writing, particularly in the context of physics, science education, and the philosophy of science. It frequently activates on words like "universities", "science", "class", "theories", and other academic terminology. The neuron may be involved in recognizing and generating text related to scientific discourse and academic writing.

5. This neuron appears to activate on scientific and mathematical notation, particularly superscripts, subscripts, and special characters used in equations and formulas. It may play a role in processing and understanding technical or scientific text.

7. The token "by" often appears before introducing a variable, parameter, or label in mathematical or scientific text. It is frequently used to define or denote specific elements in equations, models, or experimental setups.

8. The neuron appears to activate on numerical digits, particularly the digit "4", within scientific or technical contexts such as citations, measurements, or equipment specifications. This suggests the neuron may play a role in identifying or processing numerical information in academic or technical writing.

9. The token "," after various phrases in scientific or technical writing, often used to separate clauses or elements in a list. This neuron may be detecting punctuation patterns in formal, academic-style text.

10. This neuron appears to activate on abbreviations and short identifiers in academic or scientific references, particularly those related to publications, databases, or online resources. Examples include "cites", "NY", "ZIN", "TER", "SI", "e-", "cond", "Compustat", "ASP", "IN", "CAS", "Physics", "Pren", "ourworld", "compuserve", and "APS". These often appear in bibliographic entries, URLs, or other citation-related contexts in academic writing.

Specialized SAE with Tilt 500

0. The token "0" appearing in scientific notation, particularly in journal citations, volume numbers, and page numbers. This neuron may be involved in recognizing and processing numerical information in academic or scientific contexts.

5. This neuron appears to activate on mathematical and scientific notation, particularly equations, variables, and symbols. It seems to be sensitive to complex mathematical expressions, physical constants, and scientific formulas across various fields including physics, chemistry, and engineering. The neuron may play a role in processing and generating technical scientific content.

7. The neuron appears to activate on punctuation marks, particularly commas and angle brackets, when used to separate or enclose items in mathematical or scientific notation. It may play a role in parsing and understanding the structure of technical or mathematical text.

9. The token "," after phrases or clauses, often used to separate elements in scientific or technical writing. This neuron may be detecting punctuation patterns in formal, academic text.

F Automated Interpretability Prompts

In this section, we present the Interpreter and Predictor prompts used in our automated interpretation process.

F.1 Interpreter Prompt

The Interpreter prompt is designed to analyze neuron activations and explain what causes a specific neuron to activate. It is given a list of text examples where the neuron activates, with the activating tokens highlighted.

Interpreter Prompt

```
SYSTEM = """You are a meticulous AI researcher conducting
an important investigation into a certain neuron in a
language model. Your task is to analyze the neuron and
explain what causes the neuron to activate.
{prompt}
Guidelines:
You will be given a list of text examples on which the
neuron activates. The specific tokens which cause the
neuron to activate will appear between delimiters like
<<this>>. If a sequence of consecutive tokens all cause
the neuron to activate, the entire sequence of tokens
will be contained between delimiters <<just like this>>.
- You must produce a concise final description. Simply
  describe the text features that activate the neuron,
  and what its role might be based on the tokens it
  predicts.
- The last line of your response must be the formatted
  explanation.
- Think carefully about the patterns in the text examples
  and the tokens that activate the neuron. Pay attention
  to detail.
{subject_specific_instructions}"""
```

F.1.1 Example Application of Interpreter Prompt

Here's an example of how the Interpreter prompt is applied:

Interpreter Example

```
EXAMPLE_1 = """
Example 1:  and he was <<over the moon>> to find
Example 2:  we'll be laughing <<till the cows come home>>! Pro
Example 3:  thought Scotland was boring, but really there's more
<<than meets the eye>>! I'd
"""
EXAMPLE_1_EXPLANATION = """
[EXPLANATION]: Common idioms in text conveying positive sentiment.
"""
```

F.2 Predictor Prompt

The Predictor prompt is used to determine whether given text examples possess a specific linguistic feature. It returns a binary classification for each example.

Predictor Prompt

```
DSCORER_SYSTEM_PROMPT = """You are an intelligent and
meticulous linguistics researcher.
You will be given a certain feature of text, such as
"male pronouns" or "text with negative sentiment".
You will then be given several text examples. Your task
is to determine which examples possess the feature.
For each example in turn, return 1 if the sentence is
correctly labeled or 0 if the tokens are mislabeled. You
must return your response in a valid Python list. Do not
return anything else besides a Python list.
"""
```

F.2.1 Example Application of Predictor Prompt

Here's an example of how the Predictor prompt is applied:

Predictor Example

```
DSCORER_EXAMPLE_1 = """Feature explanation: "of" before words that start
with a capital letter.
Text examples:
Example 0: climate, Tomblinâ Chief of Staff Charlie Lorensen said.
Example 1: no wonderworking relics, no true Body and Blood of Christ,
no true Baptism
Example 2:Deborah Sathe, Head of Talent Development and Production
at Film London,
Example 3: It has been devised by Director of Public Prosecutions (DPP)
Example 4: and fair investigation not even include the Director of
Athletics? Finally, we believe the
"""
DSCORER_RESPONSE_1 = "[1,1,1,1,1]"
```