Steering semantic search with interpretable features from sparse autoencoders

Anonymous Author(s)
Affiliation
Address
email

Abstract

Modern information retrieval systems increasingly rely on dense neural vector embeddings, but dense embeddings of text are inherently difficult to interpret and steer, leading to opaque and potentially biased results. Sparse autoencoders (SAEs) have previously shown promise in extracting interpretable features from complex neural networks. In this work, we present the application of SAEs to dense text embeddings from large language models, demonstrating their effectiveness in disentangling document-level semantic concepts. By training SAEs on embeddings of over 420,000 scientific paper abstracts from computer science and astronomy, we show that the resulting sparse representations maintain semantic fidelity while offering high levels of interpretability. In the context of a semantic search system for scientific literature, we demonstrate that interpretable SAE features can be used to precisely steer information retrieval, allowing for fine-grained modifications of queries. At a given fidelity level to the original query, SAE feature interventions can be interpreted with $\sim$10% higher accuracy, while maintaining overall quality of information retrieval. We open source our embeddings, trained sparse autoencoders, and interpreted features, as well as a web app for exploring them.

1 Introduction

Dense vector embeddings capture nuanced semantic relationships, enabling powerful semantic search (Reimers et al., 2019; Gao et al., 2022; Wang et al., 2024; Devlin et al., 2018; Brown et al., 2020). However, the power of these representations comes at a cost: reduced interpretability and limited user control, presenting significant challenges for fine-tuning and explaining search results (Liu et al., 2019; Turian et al., 2010; Cao et al., 2023). Interpretability and intervention methods are thus unable to fully address the societal biases exhibited in the generations and representations of modern language models (Hofmann et al., 2024; Bolukbasi et al., 2016).

Sparse autoencoders (SAEs) have emerged as a promising solution for extracting interpretable features from high-dimensional representations (Ng et al., 2011; Makhzani et al., 2013). SAEs have shown success in interpreting and steering the generation outputs of diffusion models and decoder-only transformers (Conmy et al., 2024; Lee, 2024; Cunningham et al., 2023b; Elhage et al., 2022b; Daujotas, 2024), but their application to dense text embeddings remains unexplored.

In this work, we demonstrate how SAE features derived from dense text embeddings can be used to steer semantic search. By causally manipulating features in the SAE hidden dimension, we can precisely adjust the semantic meaning of queries. Our research makes the following key contributions:

1. We train varying-size SAEs on embeddings from a large corpus of scientific papers, demonstrating their effectiveness in learning interpretable features from dense text representations.

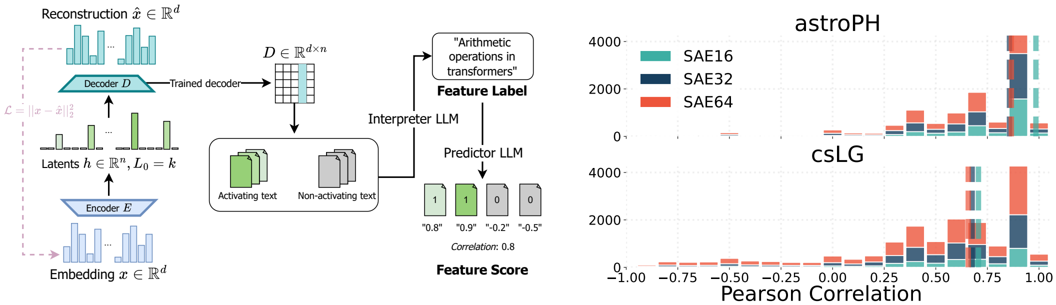


Figure 1: Left: sparse autoencoder training and labelling process. Right: interpretability of features.

2. We demonstrate the practical utility of interpretable features in enhancing semantic search, allowing fine-grained control over query semantics. We develop and open-source a tool that implements our SAE-enhanced semantic search system, as well as the underlying models.

2 Related work

Dense embeddings for text The evolution from simple one-hot encodings to sophisticated dense vector embeddings has offered substantial improvements in semantic expressiveness and contextual understanding, from Word2Vec (Mikolov et al., 2013a) and GloVe (Pennington et al., 2014), to ELMo (Peters et al., 2018) and BERT (Devlin et al., 2018), and most recently sentence-level embeddings such as Sentence-BERT (Reimers et al., 2019). Semantic search with dense embeddings has largely replaced traditional keyword search (Gao et al., 2021; Manning et al., 2008; Baeza-Yates et al., 1999; Furnas et al., 1987; Mikolov et al., 2013b; Devlin et al., 2018; Reimers et al., 2019). However, the opacity of dense embeddings can be particularly problematic in applications where explainability or precise semantic control is critical, particularly in search results.

Sparse autoencoders Sparse representations of text are often more interpretable (Trifonov et al., 2018). However, in large language models, the superposition hypothesis suggests that dense neural networks are highly underparameterised, and perform computations involving many more concepts than neurons by representing many sparse concepts, or *features*, in dense superposition (Elhage et al., 2022a). Distributed representations allows models to efficiently encode a large number of features in a relatively low-dimensional space, but it also makes model layers challenging to interpret directly. Sparse autoencoders (SAEs) address this by learning to reconstruct inputs using a sparse set of features in a higher-dimensional space, encouraging disentanglement of distributed representations (Elhage et al., 2022b; Donoho, 2006; Olshausen et al., 1997). When applied to language model activations, SAEs recover semantically meaningful and human-interpretable sparse features (Gao et al., 2024; Bricken et al., 2023; Cunningham et al., 2023b). A number of approaches for automated feature interpretation have been proposed, such as Bills et al. (2023) and Foote et al. (2023).

Activation Steering and Causal Intervention Activation steering – modifying model activations to influence downstream behavior – has emerged as a promising approach to enhance the controllability of semantic search (Li et al., 2024; Turner et al., 2023; Radford et al., 2015). Recent advancements have leveraged sparse autoencoders to identify interpretable features for precise semantic edits (Lee, 2024; Conmy et al., 2024). The field has expanded to include concept scrubbing (Belrose et al., 2024) and broader representation engineering (Zhao et al., 2024), underpinned by theoretical work on activation space geometry (Marks et al., 2023) and superposition in neural networks (Elhage et al., 2022a). Recent studies (Chan et al., 2022; Hase et al., 2023) have empirically analyzed the efficacy of causal interventions.

3 Training SAEs and automated labelling

Architecture and objective: Let $\mathbf{x} \in \mathbb{R}^d$ be an input vector, and $\mathbf{h} \in \mathbb{R}^n$ be the hidden representation, where typically $n \gg d$. The encoder and decoder functions are defined as

Encoder : $\mathbf{h} = f_\theta(\mathbf{x}) = \sigma(W_e\mathbf{x} + \mathbf{b}_e)$ and Decoder : $\hat{\mathbf{x}} = g_\phi(\mathbf{h}) = W_d\mathbf{h} + \mathbf{b}_d$ where $W_e \in \mathbb{R}^{n\times d}$ and $W_d \in \mathbb{R}^{d\times n}$ are the encoding and decoding weight matrices, $\mathbf{b}_e \in \mathbb{R}^k$ and $\mathbf{b}_d \in \mathbb{R}^d$ are bias vectors, and $\sigma(\cdot)$ is a non-linear activation function. We minimize $\mathcal{L}(\theta,\phi) = \frac{1}{d}\|\mathbf{x} - \hat{\mathbf{x}}\|_2^2 + \alpha\mathcal{L}_{\text{aux}}(\mathbf{x}, \hat{\mathbf{x}})$. Instead of an L1 penalty, we use a k-sparse constraint (Makhzani et al., 2013; Gao et al., 2024). We employ an auxiliary loss inspired by "ghost grads" (Jermyn et al., 2023) to revive dead latents (inactive for ≥ 1 epoch) and enhance model capacity; $\mathcal{L}_{\text{aux}}(\mathbf{x}, \hat{\mathbf{x}}) = |\mathbf{e} - \hat{\mathbf{e}}|_2^2$ where $\mathbf{e} = \mathbf{x} - \hat{\mathbf{x}}$ is the model residual, and $\hat{\mathbf{e}} = W_d\mathbf{z}$ is a reconstruction using dead latents; more details in Appendix A.

Training: We train two sets of SAEs on abstract embeddings from arXiv's `astro-ph` (astrophysics, 272,000), and `cs.LG` tag (computer science, 153,000). Embeddings are generated from OpenAI's `text-embedding-3-small` model and normalized zero mean and unit variance. We evaluate trained SAEs using both dead latents and normalized reconstruction MSE.

Hyperparameters: We consider the active latents k, total latents n, auxiliary latents k_{aux}, learning rate, and aux-loss coefficient α. Learning rate (set to 1e-4) and α (set to 1/32) had minimal impact on reconstruction loss. We vary k (16-128) and n (2-9 times d_{input}), training models for $\sim$13.2k steps.

Automated interpretability: To interpret features, we use two LLM instances: the Interpreter and Predictor. The Interpreter generates feature labels based on top-activating and non-activating abstracts. The Predictor uses the label to predicting activation likelihood on new abstracts, from -1 to +1. We measure the Pearson correlation between this score and true activation, and calculate the F1 score for binary classification. We use `gpt-4o` as the Interpreter and `gpt-4o-mini` as the Predictor, predicting each abstract separately; see Appendix B for more details.

4 Evaluating effectiveness of search interventions

Intervening on embeddings with SAE features SAEs are inherently correlational; however, Bricken et al. (2023), Cunningham et al. (2023a) and others demonstrate that many SAE features also have downstream causal effects. To intervene on an embedding along an SAE feature direction, we directly manipulate features in the SAE hidden dimension, and decode the result. As an implementation detail, we note that intervening on a feature by up- or down-weighting its hidden representation and then decoding is equivalent to directly adding the scaled feature vector to the final embedding. This capability is demonstrated in our open-source semantic search tool (see Appendix D). We also explore an alternative process in Appendix C where we iteratively optimise the encoded decoded latents to minimise the difference between the desired feature activations and the actual activations.

Experiment setup We incorporate SAE-based embedding interventions into a literature retrieval system for `cs.LG` and `astro-ph`. To assess the effectiveness of SAE feature intervention on semantic search, we evaluate the *specificity* and *interpretability* of feature-centric query modifications. We select random samples ($N = 50$ each) real literature retrieval queries relevant to machine learning and astronomy, which are answerable with information in papers from `cs.LG` and `astro-ph`. For each query, we return the top $k = 10$ most relevant papers using embedding cosine similarity, making up the original retrieval results $\mathcal{R}$. We then select a random feature i in the top-k from the query's hidden representation $\mathbf{h_q}$, and another orthogonal feature j that has no overlap with the top-k; we limit our selection only to features that are highly interpretable (F1 > 0.9, Pearson > 0.9). Given these features, we create a modified query embedding with $\mathbf{h}'_{\mathbf{q},\mathbf{i}} = \lambda_-$ and $\mathbf{h}'_{\mathbf{q},\mathbf{j}} = \lambda_+$, letting $\lambda_- = 0$ and sampling $\lambda_+ \in [0, 5]$. This effectively "down-weights" and "up-weights" the importance of i and j, respectively, in the modified query, which is used to generate new retrieval results $\mathcal{R}'$.

To the effect of up-weighting and down-weighting query modifications on the end retrieval results, we provide both $\mathcal{R}$ and $\mathcal{R}'$ to an external LLM instance. The external LLM then compares $\mathcal{R}$ and $\mathcal{R}'$ and determines which features, out of a multiple-choice subset of 5 options, have been up-weighted or down-weighted; we use this to compute the intervention accuracy, which measures the precision and efficacy of causal query interventions. As a baseline, we compare our SAE-based method against traditional query rewriting, by using another LLM instance to re-write the original query such that it up-weights j and down-weights i entirely using natural language.

Intervention results Our results are shown in Figure 2. We find that SAE feature interventions consistently outperform traditional query rewriting across various levels of query fidelity. This Pareto improvement demonstrates that our method can achieve higher intervention accuracy while

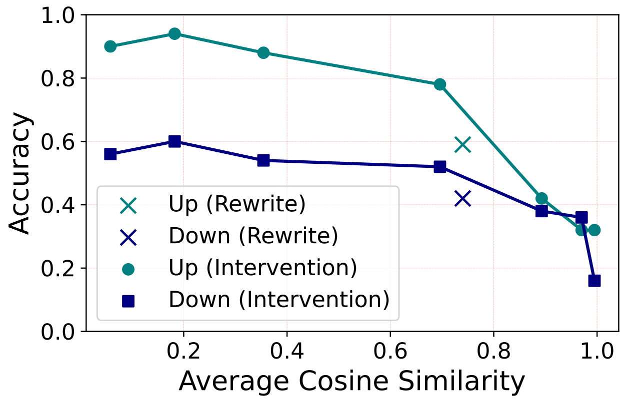


Figure 2: Relationship between intervention accuracy and query fidelity for SAE-based embedding interventions versus traditional query rewriting in literature retrieval for `cs.LG` and `astro-ph` domains. Intervention accuracy measures the precision of causal query modifications, while query fidelity is quantified by cosine similarity between original and modified query embeddings.

maintaining greater similarity to the original query. For instance, at a cosine similarity of 0.75, SAE interventions achieve approximately 10% higher accuracy compared to query rewriting.

5 Discussion

In this work, we have presented the first application of sparse autoencoders (SAEs) to semantic search using dense text embeddings. By training SAEs on embeddings of scientific paper abstracts, we have shown their effectiveness in disentangling interpretable semantic concepts in document-level embeddings. We also designed and performed a causal intervention experiment to compare the efficacy of SAE feature manipulations and direct query rewriting, demonstrating that SAE-based manipulation can precisely and interpretably steer semantic search.

While our current SAEs are trained on narrow scientific domains, extending this to the entirety of arXiv or even internet-scale text corpora could yield general-purpose SAEs with exceptionally rich feature spaces. By providing a proof-of-concept for extracting interpretable features from dense embeddings, and using features to precisely steer semantic search, our work opens several promising research directions and applications across various NLP tasks. In classification tasks, extracting interpretable and sparse features could offer fine-grained insights into model decision boundaries with global features. For machine translation, causal interventions along gender-based features could enable targeted semantic manipulations, potentially addressing issues like gender bias in translations (Stanovsky et al., 2019; Bolukbasi et al., 2016). Similar interventions could be applied to decrease bias and toxicity in the outputs of semantic search systems or generative models. Beyond these applications, our work supports the broader goal of making language models more transparent and controllable, which is crucial for building trust in AI systems as they become more integrated into critical decision-making processes (Doshi-Velez et al., 2017).

Limitations Our work focused on relatively small datasets from specific scientific domains. Although this specificity allowed us to demonstrate the effectiveness of our steering approach in targeted search domains, future work should investigate generalization to larger, more diverse corpora; SAEs for general text embeddings would also need to be scaled up by at least 2-3 the total number of latents. Of particular interest would be corpora and intervention experiments focused on debiasing results or decreasing toxicity in information retrieval. It would also be extremely useful to have human evaluations, in order to evaluate the end-user interpretability of our steering approach. Additionally, our automated interpretability process is correlational and does not a priori guarantee that direct manipulation of the feature aligns with the interpretation. We would also suggest future work evaluating performance of reconstructed embeddings on benchmarks like MTEB (Muennighoff et al., 2022), and comparing learned dictionaries to some proxy of ground-truth features (Makelov et al., 2024; Olah et al., 2024), in order to understand the completeness of recovered features.

References

Baeza-Yates, Ricardo, Berthier Ribeiro-Neto, et al. (1999). *Modern information retrieval*. Vol. 463. ACM press New York.

Belrose, Nora, David Schneider-Joseph, Shauli Ravfogel, Ryan Cotterell, Edward Raff, and Stella Biderman (2024). "Leace: Perfect linear concept erasure in closed form". In: *Advances in Neural Information Processing Systems* 36.

Bills, Steven, Nick Cammarata, Dan Mossing, Henk Tillman, Leo Gao, Gabriel Goh, Ilya Sutskever, Jan Leike, Jeff Wu, and William Saunders (2023). "Language models can explain neurons in language models". In: *URL https://openaipublic. blob. core. windows. net/neuron-explainer/paper/index. html.(Date accessed: 14.05. 2023)* 2.

Bolukbasi, Tolga, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai (2016). "Man is to computer programmer as woman is to homemaker? debiasing word embeddings". In: *Advances in neural information processing systems*, pp. 4349–4357.

Bricken, Trenton, Catherine Olsson, and Neel Nanda (2023). "Towards Monosemanticity: Decomposing Language Models With Dictionary Learning". In: *arXiv preprint arXiv:2301.05498*.

Brown, Tom, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. (2020). "Language Models are Few-Shot Learners". In: *Advances in neural information processing systems* 33, pp. 1877–1901.

Cao, Wenqiang, Qing Li, Siying Zhang, Rixin Xu, and Youqi Li (2023). "STEP: Generating Semantic Text Embeddings with Prompt". In: *2023 Eleventh International Conference on Advanced Cloud and Big Data (CBD)*, pp. 180–185. URL: https://api.semanticscholar.org/CorpusID:269628678.

Chan, Lawrence, Adria Garriga-Alonso, Nicholas Goldowsky-Dill, Ryan Greenblatt, Jenny Nitishinskaya, Ansh Radhakrishnan, Buck Shlegeris, and Nate Thomas (2022). "Causal scrubbing: A method for rigorously testing interpretability hypotheses". In: *AI Alignment Forum*, p. 10.

Conmy, Arthur and Neel Nanda (2024). *Activation Steering with SAEs*. Accessed 16-07-2024. URL: https://www.lesswrong.com/posts/C5KAZQib3bzzpeyrg/full-post-progress-update-1-from-the-gdm-mech-interp-team#Activation_Steering_with_SAEs.

Cunningham, Hoagy, Aidan Ewart, Logan Riggs, Robert Huben, and Lee Sharkey (2023a). *Sparse Autoencoders Find Highly Interpretable Features in Language Models*. arXiv: 2309.08600 [cs.LG]. URL: https://arxiv.org/abs/2309.08600.

– (2023b). "Sparse autoencoders find highly interpretable features in language models". In: *arXiv preprint arXiv:2309.08600*.

Daujotas, Gytis (2024). *Interpreting and Steering Features in Images*. https://www.lesswrong.com/posts/Quqekpvx8BGMMcaem/interpreting-and-steering-features-in-images. [Accessed 16-07-2024].

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2018). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *arXiv preprint arXiv:1810.04805*.

Donoho, David L (2006). "Compressed sensing". In: *IEEE Transactions on Information Theory* 52.4, pp. 1289–1306.

Doshi-Velez, Finale and Been Kim (2017). *Towards A Rigorous Science of Interpretable Machine Learning*. arXiv: 1702.08608 [stat.ML]. URL: https://arxiv.org/abs/1702.08608.

Elhage, Nelson, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg, and Christopher Olah (2022a). *Toy Models of Superposition*. arXiv: 2209.10652 [cs.LG]. URL: https://arxiv.org/abs/2209.10652.

Elhage, Nelson, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Johnston, Ben Mann, Amanda Askell, Danny Hernandez, Dawn Drain, Zac Hatfield-Dodds, et al. (2022b). "Softmax Linear Units". In.

Foote, Alex, Neel Nanda, Esben Kran, Ioannis Konstas, Shay Cohen, and Fazl Barez (2023). "Neuron to graph: Interpreting language model neurons at scale". In: *arXiv preprint arXiv:2305.19911*.

Furnas, George W, Thomas K Landauer, Louis M Gomez, and Susan T Dumais (1987). "The vocabulary problem in human-system communication". In: *Communications of the ACM* 30.11, pp. 964–971.

Gao, Leo, John Thickstun, Anirudh Madaan, Zach Scherlis, Arush Guha, Sumanth Dathathri, Jared Kaplan, Azalia Mirhoseini, and Ilya Sutskever (2024). "Scaling Laws for Neurons in GPT Models". In: *arXiv preprint arXiv:2401.02325*.

Gao, Luyu, Xueguang Ma, Jimmy Lin, and Jamie Callan (2022). *Precise Zero-Shot Dense Retrieval without Relevance Labels*. arXiv: 2212.10496 [cs.IR]. URL: https://arxiv.org/abs/2212.10496.

Gao, Tianyu, Xingcheng Yao, and Danqi Chen (2021). "SimCSE: Simple contrastive learning of sentence embeddings". In: *arXiv preprint arXiv:2104.08821*.

Hase, Peter, Mohit Bansal, Been Kim, and Asma Ghandeharioun (2023). *Does Localization Inform Editing? Surprising Differences in Causality-Based Localization vs. Knowledge Editing in Language Models*. arXiv: 2301.04213 [cs.LG]. URL: https://arxiv.org/abs/2301.04213.

Hofmann, Valentin, Pratyusha Ria Kalluri, Dan Jurafsky, and Sharese King (2024). "AI generates covertly racist decisions about people based on their dialect". In: *Nature* 615, pp. 78–85. DOI: 10.1038/s41586-024-07856-5. URL: https://www.nature.com/articles/s41586-024-07856-5.

Jermyn, Adam and Adly Templeton (2023). *Ghost Grads: An improvement on resampling*. [Accessed 19-07-2024]. URL: https://transformer-circuits.pub/2024/jan-update/index.html#dict-learning-resampling.

Kingma, Diederik P and Jimmy Ba (2014). "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980*.

Lee, Linus (2024). *Prism: mapping interpretable concepts and features in a latent space of language*. Accessed 16-07-2024. URL: https://thesephist.com/posts/prism.

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

Liu, Nelson F, Matt Gardner, Yonatan Belinkov, Matthew E Peters, and Noah A Smith (2019). "Linguistic knowledge and transferability of contextual representations". In: *arXiv preprint arXiv:1903.08855*.

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

Makhzani, Alireza and Brendan Frey (2013). "K-sparse autoencoders". In: *arXiv preprint arXiv:1312.5663*.

Manning, Christopher D, Prabhakar Raghavan, and Hinrich Schütze (2008). *Introduction to information retrieval*. Cambridge university press.

Marks, Samuel and Max Tegmark (2023). "The geometry of truth: Emergent linear structure in large language model representations of true/false datasets". In: *arXiv preprint arXiv:2310.06824*.

Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean (2013a). "Efficient estimation of word representations in vector space". In: *arXiv preprint arXiv:1301.3781*.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean (2013b). "Distributed representations of words and phrases and their compositionality". In: *Advances in neural information processing systems* 26.

Muennighoff, Niklas, Nouamane Tazi, Loic Magne, and Nils Reimers (2022). "MTEB: Massive text embedding benchmark". In: *arXiv preprint arXiv:2210.07316*.

Ng, Andrew et al. (2011). "Sparse autoencoder". In: *CS294A Lecture notes*. Vol. 72. 2011, pp. 1–19.

Olah, Chris and Adam Jermyn (2024). *July Update*. https://transformer-circuits.pub/2024/july-update/. URL: https://transformer-circuits.pub/2024/july-update/.

Olshausen, Bruno A and David J Field (1997). "Sparse coding with an overcomplete basis set: A strategy employed by V1?" In: *Vision Research* 37.23, pp. 3311–3325.

Pennington, Jeffrey, Richard Socher, and Christopher D Manning (2014). "Glove: Global vectors for word representation". In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543.

Peters, Matthew E, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer (2018). "Deep contextualized word representations". In: *arXiv preprint arXiv:1802.05365*.

Radford, Alec, Luke Metz, and Soumith Chintala (2015). "Unsupervised representation learning with deep convolutional generative adversarial networks". In: *arXiv preprint arXiv:1511.06434*.

Reimers, Nils and Iryna Gurevych (2019). "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks". In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, pp. 3982–3992.

Stanovsky, Gabriel, Noah A Smith, and Luke Zettlemoyer (2019). “Evaluating Gender Bias in Machine Translation”. In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Trifonov, Valentin, Octavian-Eugen Ganea, Anna Potapenko, and Thomas Hofmann (2018). *Learning and Evaluating Sparse Interpretable Sentence Embeddings*. arXiv: 1809.08621 [cs.CL]. URL: https://arxiv.org/abs/1809.08621.

Turian, Joseph, Lev Ratinov, and Yoshua Bengio (2010). “Word representations: a simple and general method for semi-supervised learning”. In: *Proceedings of the 48th annual meeting of the association for computational linguistics*, pp. 384–394.

Turner, Alex, Lisa Thiergart, David Udell, Gavin Leech, Ulisse Mini, and Monte MacDiarmid (2023). “Activation addition: Steering language models without optimization”. In: *arXiv preprint arXiv:2308.10248*.

Wang, Liang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei (2024). *Text Embeddings by Weakly-Supervised Contrastive Pre-training*. arXiv: 2212.03533 [cs.CL]. URL: https://arxiv.org/abs/2212.03533.

Zhao, Shuai, Meihuizi Jia, Luu Anh Tuan, Fengjun Pan, and Jinming Wen (2024). “Universal vulnerabilities in large language models: Backdoor attacks for in-context learning”. In: *arXiv preprint arXiv:2401.05949*.

Contents

A Training details

A.1 Training setup

Our sparse autoencoder (SAE) implementation incorporates several recent advancements in the field. Following Bricken et al. (2023), we initialise the bias b_{pre} using the geometric median of a data point sample and set encoder directions parallel to decoder directions. Decoder latent directions are normalised to unit length at initialisation and after each training step. For our top-k models, based on Gao et al. (2024), we set initial encoder magnitudes to match input vector magnitudes, though our analyses indicate minimal impact from this choice.

We augment the primary loss with an auxiliary component (AuxK), inspired by the "ghost grads" approach of Jermyn et al. (2023). This auxiliary term considers the top-k_{aux} inactive latents (typically $k_{aux} = 2k$), where inactivity is determined by a lack of activation over a full training epoch. The total loss is formulated as $\mathcal{L} + \alpha\mathcal{L}_{aux}$, with α usually set to 1/32. This mechanism reduces the number of dead latents with minimal computational overhead (Gao et al., 2024). We found that dead latents only occurred during training the $k = 16$ models, and all dead latents had disappeared by the end of training. We show how dead latents evolved over training the $k = 16$ SAEs for the `astro-ph` abstracts in Figure 3.

For optimisation, we employ Adam (Kingma et al., 2014) with $\beta_1 = 0.9$ and $\beta_2 = 0.999$, maintaining a constant learning rate. We use gradient clipping. Our training uses batches of 1024 abstracts, with

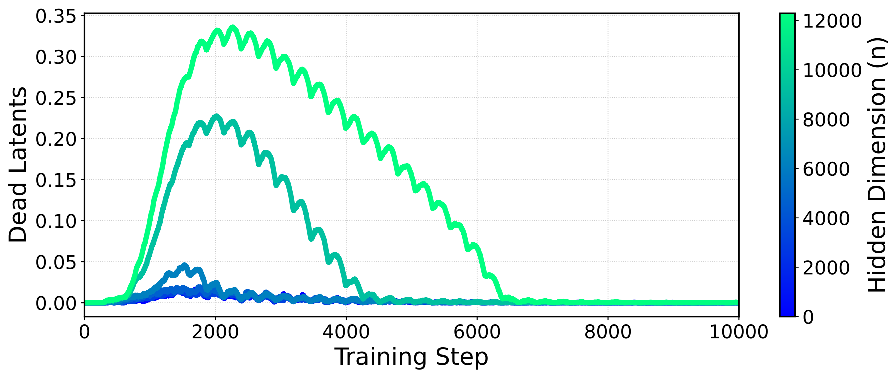


Figure 3: The proportion of dead latents, defined as features that haven't fired in the last epoch of training, for our $k = 16$ SAEs on the `astro-ph` abstract embeddings. All dead latents were gone by the end of training. We found that dead latents only occurred in $k = 16$ autoencoders.

performance metrics showing robustness to batch size variations under appropriate hyperparameter settings.

The primary MSE loss uses a global normalisation factor computed at training initiation, while the AuxK loss employs per-batch normalisation to adapt to evolving error distributions. Following Bricken et al. (2023), we apply a gradient projection technique to mitigate interactions between the Adam optimiser and decoder normalisation.

A.2 SAE training metrics

Table 1 shows the final training metrics for all combinations of SAEs trained. We note clear trends in normalised MSE, log feature density and activation mean as we vary the number of active latents k and the overall number of latents n.

A.3 Interpretability of SAE features

The most direct way to evaluate the interpretability of features is to look at the distribution of automated interpretability scores, discussed above. Specifically: given a feature label from our interpreter model, how well can a predictor model predict the feature's activation on unseen text? We show in Figure 4 that the Pearson correlation between predictor model confidence of a feature firing and the ground-truth firing is quite high, with median correlations ranging from 0.65 to 0.71 for `cs.LG` and 0.85 to 0.98 for `astro-ph`. We note that Pearson correlation increases as k and n decrease, likely due to models learning coarser-grained features that are easier for the interpreter to identify.

B Automated interpretability details

B.1 Examples of features

We show some examples of perfectly interpretable features (Pearson correlation > 0.99) in Table 2. The strength of the activation of the feature on its top 3 activating abstracts is shown in parentheses next to the abstract title.

B.2 Exploring the effectiveness of smaller models

Although we eventually used `gpt-4o-mini` as the Predictor model, we initially did some ablations to understand how effective `gpt-4o` and `gpt-3.5-turbo` would be as different combinations of the Interpreter and Predictor models. We measured this by randomly sampling 50 features from our SAE64 (trained on `astro-ph` abstracts) and measuring the interpretability scores of different model combinations, in terms of both F1 score (does the model's binary classification of a feature firing on an abstract agree with the ground-truth) and the Pearson correlation (described in the main body).

Table 1: Metrics for our top-k sparse autoencoders with varying k and hidden dimensions, across both astronomy and computer science papers. MSE is normalised mean squared error, Log FD is the mean log density of feature activations, and activation mean is the mean activation value across non-zero features. Note that MSE is normalised.

		`astro.ph`			`cs.LG`		
k	n	MSE	Log FD	Act Mean	MSE	Log FD	Act Mean
16	3072	0.2264	-2.7204	0.1264	0.2284	-2.7314	0.1332
	4608	0.2246	-4.7994	0.1350	0.2197	-3.0221	0.1338
	6144	0.2128	-3.1962	0.1266	0.2089	-3.2299	0.1342
	9216	0.1984	-3.4206	0.1264	0.1962	-3.4833	0.1343
	12288	0.1957	-6.2719	0.1274	0.1897	-3.6448	0.1347
32	3072	0.1816	-2.3389	0.0847	0.1831	-2.3008	0.0885
	4608	0.1691	-3.6091	0.0882	0.1697	-2.5152	0.0876
	6144	0.1604	-2.7761	0.0841	0.1641	-2.6687	0.0873
	9216	0.1554	-3.0227	0.0842	0.1540	-2.9031	0.0875
	12288	0.1520	-4.9505	0.0843	0.1457	-3.0577	0.0877
64	3072	0.1420	-1.9538	0.0566	0.1485	-1.8875	0.0584
	4608	0.1331	-2.7782	0.0622	0.1370	-2.0637	0.0570
	6144	0.1262	-2.2828	0.0545	0.1310	-2.1852	0.0558
	9216	0.1182	-2.4682	0.0539	0.1240	-2.3536	0.0545
	12288	0.1152	-3.4787	0.0583	0.1162	-2.4847	0.0548
128	3072	0.1111	-1.8876	0.0483	0.1206	-1.5311	0.0399
	4608	0.1033	-2.1392	0.0457	0.1137	-1.6948	0.0376
	6144	0.1048	-2.2501	0.0438	0.1076	-1.8079	0.0366
	9216	0.0975	-2.5352	0.0409	0.0999	-1.9701	0.0348
	12288	0.0936	-2.7025	0.0399	0.0942	-2.0858	0.0342

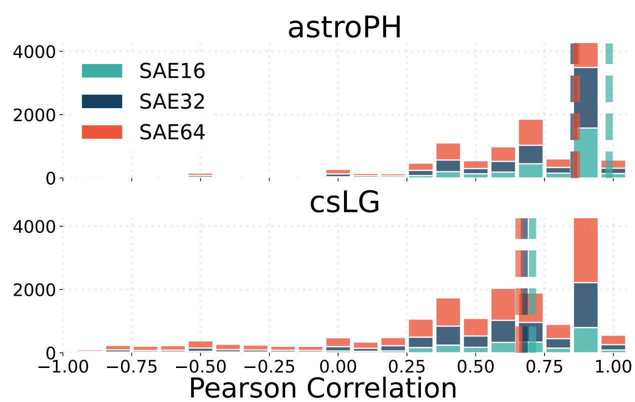


Figure 4: Pearson correlations between the ground-truth and predicted feature activation, using GPT-4o as the *Interpreter* and GPT-4o-mini as the *Predictor*.

Feature			
Astronomy			
Cosmic Microwave Background	CMB map-making and power spectrum estimation (0.1708)	How to calculate the CMB spectrum (0.1598)	CMB data analysis and sparsity (0.1581)
Periodicity in astronomical data	Generalized Lomb-Scargle analysis of decay rate measurements from the Physikalisch-Technische Bundesanstalt (0.1027)	Multicomponent power-density spectra of Kepler AGNs, an instrumental artefact or a physical origin? (0.0806)	RXTE observation of the X-ray burster 1E 1724-3045. I. Timing study of the persistent X-ray emission with the PCA (0.0758)
X-ray reflection spectra	X-ray reflection spectra from ionized slabs (0.3859)	The role of the reflection fraction in constraining black hole spin (0.3803)	Relativistic reflection: Review and recent developments in modeling (0.3698)
Critique or refutation of theories	What if string theory has no de Sitter vacua? (0.2917)	No evidence of mass segregation in massive young clusters (0.2051)	Ruling Out Initially Clustered Primordial Black Holes as Dark Matter (0.2029)
Computer Science			
Sparsity in Neural Networks	Two Sparsities Are Better Than One: Unlocking the Performance Benefits of Sparse-Sparse Networks (0.3807)	Truly Sparse Neural Networks at Scale (0.3714)	Topological Insights into Sparse Neural Networks (0.3689)
Gibbs Sampling and Variants	Herded Gibbs Sampling (0.2990)	Characterizing the Generalization Error of Gibbs Algorithm with Symmetrized KL information (0.2858)	A Framework for Neural Network Pruning Using Gibbs Distributions (0.2843)
Arithmetic operations in transformers	Arbitrary-Length Generalization for Addition in a Tiny Transformer (0.1828)	Carrying over algorithm in transformers (0.1803)	Understanding Addition in Transformers (0.1792)

Table 2: Activation strengths and titles for abstracts related to Astronomy and Computer Science features.

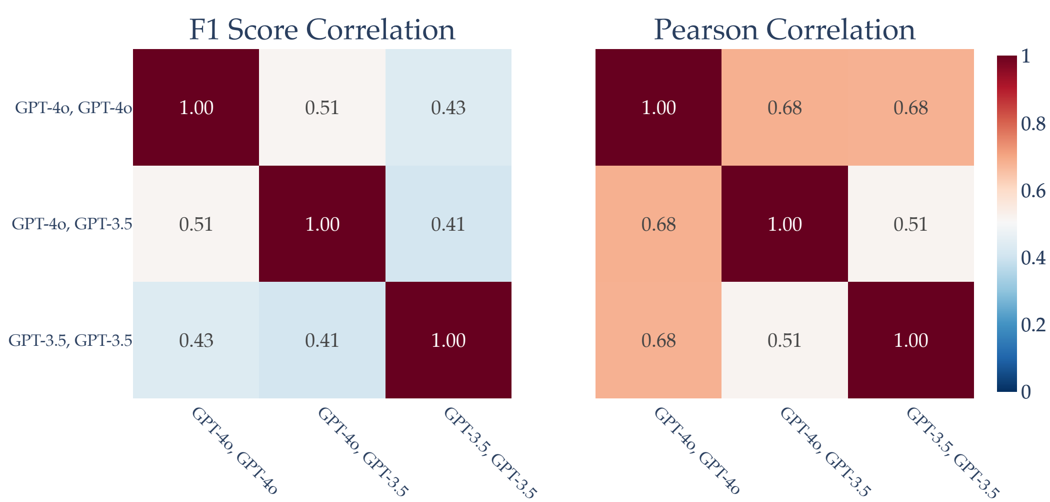


Figure 5: Correlation between F1 scores and Pearson correlation scores of different combinations of `(labeller, predictor)` models. Interestingly, using GPT-3.5 as the predictor appears to degrade performance similarly regardless of whether the feature was labelled by GPT-4o or GPT-3.5.

Interestingly, we observe that using `gpt-4o` as the Interpreter and `gpt-3.5-turbo` as the Predictor leads to similar scores as using `gpt-3.5-turbo` for both, as shown in Figures 5 and Figures 6. This suggests that the challenging task in the autointerp is not necessarily labelling but rather predicting the activation of a feature on unseen abstracts.

Another observation is that using `gpt-3.5-turbo` as the Predictor only leads to a moderate degradation of F1 score, it leads to a significant degradation of Pearson correlation. This is likely because we only use 6 abstracts for each feature prediction (3 positive, 3 negative) and thus there are only a

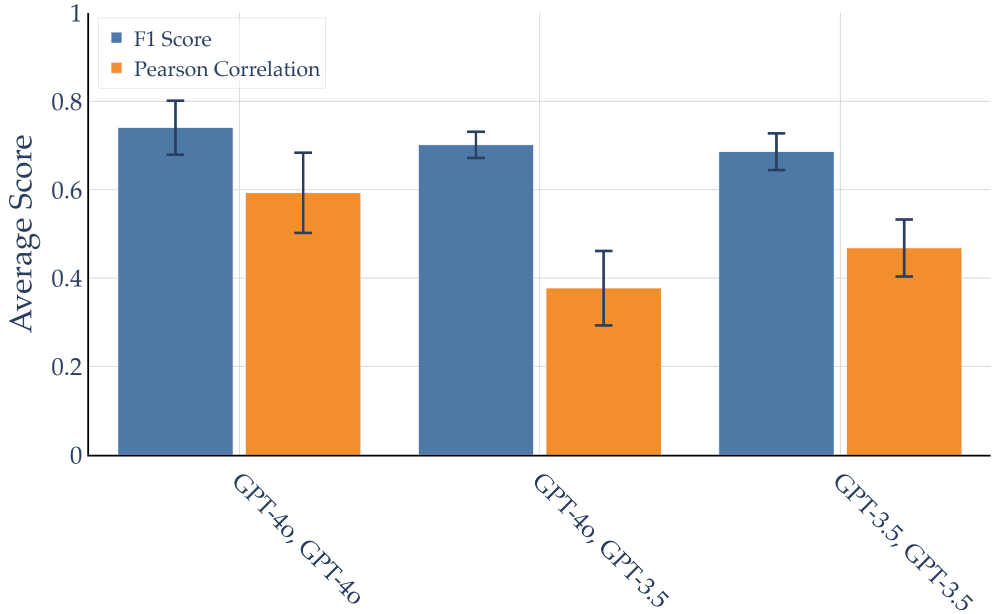


Figure 6: Mean F1 scores and Pearson correlations (according to ground-truth feature activations) across 50 randomly sampled features, for different combinations of `(Interpreter, Predictor)` models.

few discrete F1 scores possible. Additionally, it appeared that `gpt-3.5-turbo` was generally less likely to assign higher confidence scores in either direction, with a much lower variance in assigned confidence than when `gpt-4o` was the Predictor. This affects Pearson correlation but not F1.

C Iterative encoding optimisation

We noted in Section 4 that intervening on a feature by up- or down-weighting its hidden representation and then decoding is equivalent to directly adding the scaled feature vector to the final embedding. To demonstrate this equivalence, let's consider an intervention on feature i by an amount δ. The modified hidden representation is $\mathbf{h}' = \mathbf{h} + \delta \mathbf{e}_i$, where $\mathbf{e}_i$ is the i-th standard basis vector. Decoding this modified representation gives $\hat{\mathbf{x}}' = W_d \mathbf{h}' = W_d \mathbf{h} + \delta W_d \mathbf{e}_i = \hat{\mathbf{x}} + \delta \mathbf{w}_i$, where $\mathbf{w}_i$ is the i-th column of W_d. Thus, intervening on the hidden representation and then decoding is equivalent to directly adding the scaled feature vector to the original reconstruction.

We show in Figure 9 how cosine similarity between the original query embedding and the modified query embedding changes as we change the upweighting and downweighting strength for different features. Cosine similarity drops rapidly as soon as upweight or downweight exceeds 0.1.

There is an implicit challenge in SAE-based embedding interventions: the trade-off between steering strength and precision. When directly manipulating feature activations, we observed that strong interventions often led to unintended semantic shifts, activating correlated features and potentially moving the embedding far from the SAE's learned manifold. Our goal is to achieve precise semantic edits that express the desired feature strongly while minimising interference with unrelated features. To this end, we developed an iterative optimisation approach that leverages the SAE's learned feature space to find an optimal balance between these competing objectives.

Let $\mathbf{x} \in \mathbb{R}^d$ be the original embedding, $f_\theta(\cdot)$ the SAE encoder, and $g_\phi(\cdot)$ the SAE decoder. We define a target feature vector $\mathbf{t} \in \mathbb{R}^k$ representing the desired feature activations after intervention, where k is the number of active features in our SAE. The iterative latent optimisation aims to find optimised latents $\mathbf{h}^*$ that satisfy:

$$\mathbf{h}^* = \mathrm{argmin}_{\mathbf{h}'} \left\{ \| f_\theta(g_\phi(\mathbf{h}')) - \mathbf{t} \|_2^2 \right\}$$

We solve this optimisation problem using gradient descent, starting from the initial latents $\mathbf{h} = f_\theta(\mathbf{x})$ and iteratively updating $\mathbf{h}'$. We use the AdamW optimiser with a cosine annealing learning rate schedule.

To evaluate the effectiveness of this approach, we compare it to a direct intervention method where we simply set the target feature to a specific value in the latent space. For each abstract in our dataset, we embed the abstract using an OpenAI embedding model to obtain $\mathbf{x}$. We then encode the embedding to get initial latents $\mathbf{h} = f_\theta(\mathbf{x})$. We randomly select a target feature i and target value v. We then apply both intervention methods: our iterative optimisation of $\mathbf{h}'$ as described above, with $\mathbf{t}_i = v$ and $\mathbf{t}_j = \mathbf{h}_j$ for $j \neq i$, and direct intervention: setting $\mathbf{h}'_i = v$ and $\mathbf{h}'_j = \mathbf{h}_j$ for $j \neq i$.

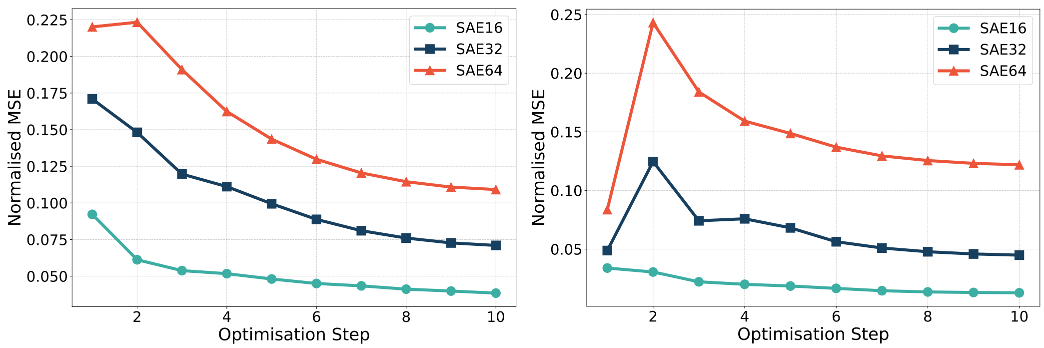


Figure 7: Normalised MSE at each of 10 steps across the iterative latent optimisation process. Left: Setting a random zero feature to active. Right: Setting a random active feature to zero.

Figure 7 (left panel) shows the trajectory of normalised MSE during the iterative optimisation process, when setting a random zero feature to active. Similarly, the right panel shows the optimisation when setting a random active feature to zero. Normalised MSE improves in the former case but not the latter.

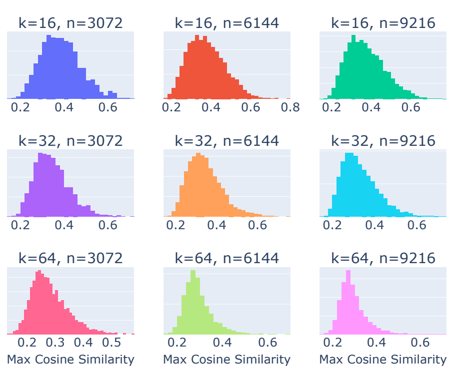


Figure 8: Distribution of maximum cosine similarity between a given feature vector and all other feature vectors, within the same SAE.

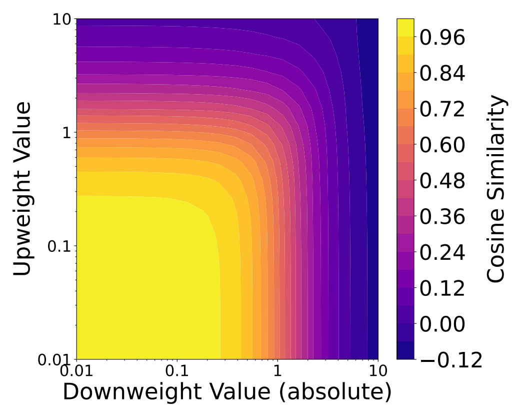


Figure 9: Cosine similarity between the original query embedding and the modified query embedding, with different values of upweighting random zero features and downweighting random active features.

D `SAErch.ai`

To demonstrate the practical applications of our sparse autoencoder (SAE) approach to semantic search and feature interpretation, we developed SAErch.ai, a web application that allows users to interact with the SAE models trained on arXiv paper embeddings.

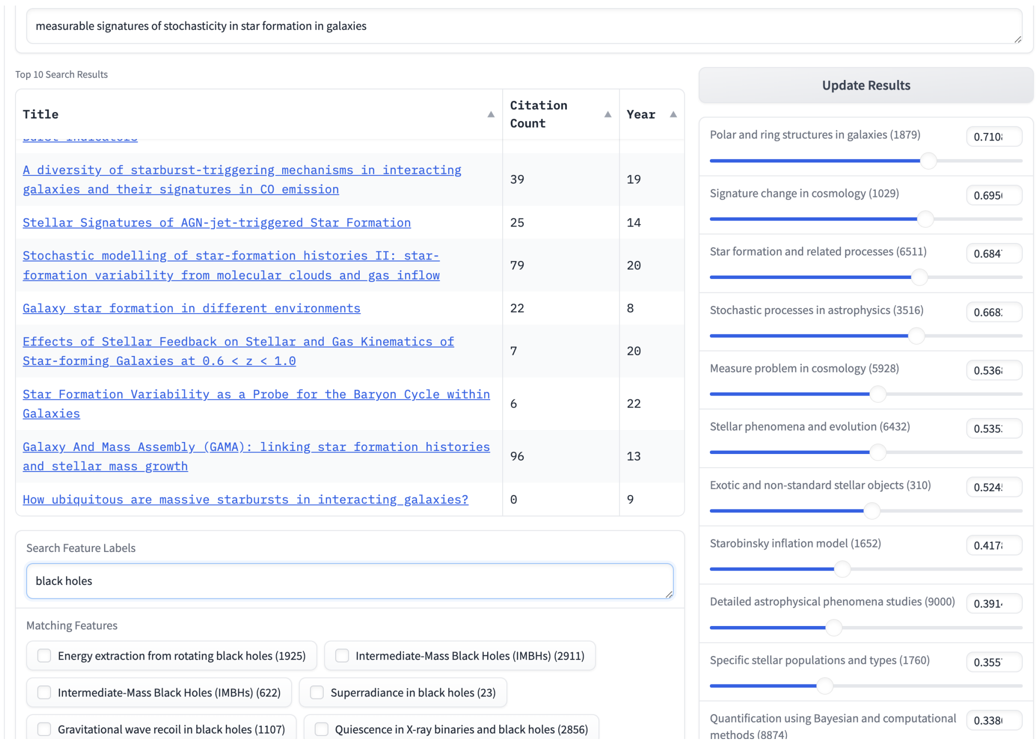


Figure 10: The SAErch tab of our web application, demonstrating a semantic search for "measurable signatures of stochasticity in star formation in galaxies" in the astrophysics domain. The interface displays the top 10 search results ranked by relevance, including title, citation count, and publication year. On the right, sliders represent the top activated SAE features for the query, allowing users to fine-tune the search by adjusting feature weights. On the bottom we have our feature addition interface. Users can search for specific semantic features (e.g., "black holes") and add them to their query. They can then adjust the strength of these features.

D.1 Overview

SAErch.ai is built using the Gradio framework and consists of three main tabs: Home, SAErch, and Feature Visualisation. The application allows users to switch between the Computer Science (`cs.LG`) and Astrophysics (`astro-ph`) datasets.

The SAErch tab implements the core functionality of our semantic search system, allowing users to:

- Input a search query
- View the top 10 search results based on embedding similarity
- Interact with the SAE features activated by their query

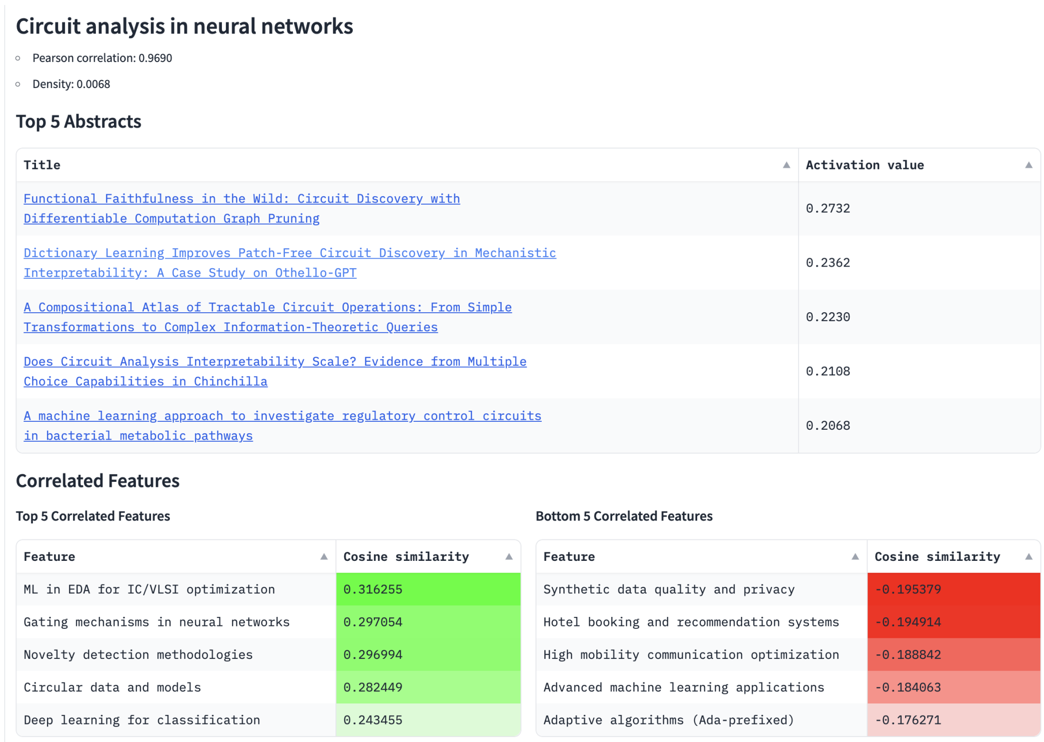
For each query, the system displays sliders corresponding to the top-k SAE features activated by the input. Users can adjust these sliders to modify the query embedding, effectively steering the search results towards or away from specific semantic concepts; see Figure 10. This directly demonstrates the fine-grained control over query semantics discussed in Section 4 of our paper. Users can also search for and add specific features not initially activated by their query.

D.2 Feature Visualisation Tab

The Feature Visualisation tab is divided into two sub-tabs: Individual Features and Feature Families.

D.2.1 Individual Features

For any selected feature, this tab displays:



Circuit analysis in neural networks

- Pearson correlation: 0.9690
- Density: 0.0068

Top 5 Abstracts

Title	Activation value
Functional Faithfulness in the Wild: Circuit Discovery with Differentiable Computation Graph Pruning	0.2732
Dictionary Learning Improves Patch-Free Circuit Discovery in Mechanistic Interpretability: A Case Study on Othello-GPT	0.2362
A Compositional Atlas of Tractable Circuit Operations: From Simple Transformations to Complex Information-Theoretic Queries	0.2230
Does Circuit Analysis Interpretability Scale? Evidence from Multiple Choice Capabilities in Chinchilla	0.2108
A machine learning approach to investigate regulatory control circuits in bacterial metabolic pathways	0.2068

Correlated Features

Top 5 Correlated Features

Feature	Cosine similarity
ML in EDA for IC/VLSI optimization	0.316255
Gating mechanisms in neural networks	0.297054
Novelty detection methodologies	0.296994
Circular data and models	0.282449
Deep learning for classification	0.243455

Bottom 5 Correlated Features

Feature	Cosine similarity
Synthetic data quality and privacy	-0.195379
Hotel booking and recommendation systems	-0.194914
High mobility communication optimization	-0.188842
Advanced machine learning applications	-0.184063
Adaptive algorithms (Ada-prefixed)	-0.176271

Figure 11: Individual feature visualisation for the "Circuit analysis in neural networks" feature in the computer science domain. The interface displays key interpretability metrics, top activating abstracts, correlated and co-occurring features, and an activation distribution histogram. Further information (not shown in the image) includes co-occurring features and activation distribution.

- Top 5 activating abstracts, demonstrating the semantic content captured by the feature
- Top and bottom 5 correlated features, illustrating the relationships between different SAE features
- Top 5 co-occurring features, showing which features tend to activate together
- A histogram of activation values, providing insight into the feature's behavior across the corpus
- The most similar features in SAE16 and SAE32

D.2.2 Feature Families

The Feature Families tab in our web application offers an in-depth exploration of related features discovered by our sparse autoencoder. We show an example feature family in Figure 12.

The table displays the parent feature (superfeature) and its child features, along with key metrics, such as the name of the parent and child features, the frequency of co-occurrence between the child feature and the parent feature, ranging from 0 to 1, and the F1 Score and Pearson correlation.

The interactive directed graph provides a visual representation of the feature family structure. Each node represents a feature. The size of the node corresponds to the feature's density (frequency of activation), while the color intensity indicates the Pearson correlation (interpretability). Arrows between nodes show relationships between features, with the direction typically pointing from more general to more specific concepts. Users can hover over nodes to view detailed information about each feature, including its name and log density.

Family and Child Features

Feature	Parent Co-Occurrence	F1 Score	Pearson
Transformer models	1	1	0.9994
Positional Encoding in Neural Networks	0.34	1	0.9992
Vision Transformers (ViT) and training techniques	0.34	1	0.9977
Theoretical and Practical Aspects of Transformers	0.24	1	0.9992
Transformer architectures and enhancements	0.72	1	0.9993
Vision Transformers and Efficiency Techniques	0.49	1	0.9923
Swin Transformer in Vision Tasks	0.49	1	1
Mamba architecture and applications	0.95	1	0.9992
Transformer architectures and applications	1	1	0.9638

Family Graph

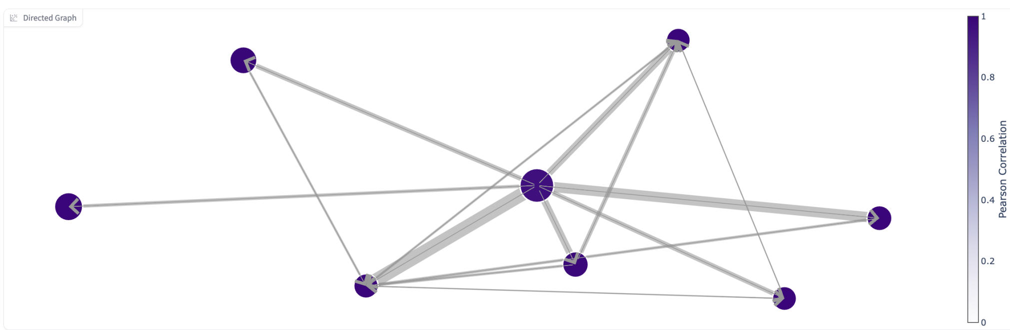


Figure 12: Directed graph visualization of a transformer models feature family. Nodes represent individual features, with size indicating feature density and color intensity showing Pearson correlation. Edges depict relationships between features, with arrow direction pointing from more general to more specific concepts. Users can hover over nodes to view detailed feature information.