ANALYSIS OF VARIATIONAL SPARSE AUTOENCODERS

ABSTRACT

Sparse Autoencoders (SAEs) have emerged as a promising approach for interpreting neural network representations by learning sparse, human-interpretable features from dense activations. We investigate whether incorporating variational methods into SAE architectures can improve feature organization and interpretability. We introduce the variational Sparse Autoencoder (vSAE), which replaces deterministic ReLU gating with stochastic sampling from learned Gaussian posteriors and incorporates KL divergence regularization toward a standard normal prior. Our hypothesis is that this probabilistic sampling creates dispersive pressure, causing features to organize more coherently in the latent space while avoiding overlap. We evaluate a Topk vSAE against a standard TopK SAE on Pythia-70M transformer residual steam activations using comprehensive benchmarks including SAE Bench, individual feature interpretability analysis, and global latent space visualization through t-SNE. The vSAE underperforms standard SAE across core evaluation metrics, though excels at feature independence and ablation metrics. The KL divergence term creates excessive regularization pressure that substantially reduces the fraction of living features, leading to observed performance degradation. While vSAE features demonstrate improved robustness, they exhibit many more dead features than baseline. Our findings suggest that naive application of variational methods to SAEs does not improve feature organization or interpretability.

***K*eywords** Mechanistic Interpretability · Sparse Autoencoders · Variational Autoencoders

1 Introduction

1.1 Background

Autoencoders: Autoencoders were first applied to neural networks in 1986 by Rumelhart, Hinton, Williams. It is an architecture that learns to compress data into a smaller representation, then reconstructs the original data from that compressed form [1]. The network has two parts: an encoder that reduces input data to a lower-dimensional latent space, and a decoder that reconstructs the input from this compressed representation. The compressed middle layer forces the network to learn meaningful patterns and remove noise. [2]

This concept was expanded upon in 2006 by Hinton and Salakhutdinov by introducing several layers, each trained in succession. Through this layer-by-layer pretraining using Restricted Boltzmann Machines, the model is "unfolded" to produce encoder and decoder networks, which is then fine tuned. It outperformed PCA, the previous best network. [3]

Variational Autoencoders: Variational autoencoders (VAEs) are probabilistic generative models that learn to encode data into a continuous latent space and decode it back to generate new samples. Unlike traditional autoencoders whose encoder maps inputs to fixed points in the latent space, encoder of the VAE instead produces two vectors: one for means and one for standard deviations, defining probability distributions for each latent dimension.

The foundational framework for VAEs was established by Kingma and Welling in their seminal 2013 paper "Auto-Encoding Variational Bayes." This work introduced the critical reparameterization trick that made VAE training feasible through standard backpropagation. This insight allows end-to-end optimization of both the encoder and decoder simultaneously. The objective function of this initial VAE naturally decomposes into two interpretable terms: a KL divergence regularizer that constrains the learned latent distributions to remain close to a simple prior (typically standard Gaussian), and a reconstruction term that ensures the decoder can faithfully regenerate the input data. [4]

Empirically, it is observed that autoencoders exhibit a non-smooth latent structure, while VAEs tend to have a smooth latent structure Latent Space Characterization of Autoencoder Variants. Research demonstrates that the latent spaces of standard autoencoders form non-smooth manifolds, while that of VAE forms a smooth manifold Latent Space Characterization of Autoencoder Variants. [5]

VAEs introduce a probabilistic structure to the latent space by learning a distribution over the latent variables rather than representing them as fixed points. This pushes the latent space toward continuous and smooth representations, facilitating tasks like data generation and interpolation with increased

Sparse Autoencoders: One of the main difficulties in mechanistic interpretability is the problem of polysemanticity of neurons within language models. Each neuron can represent multiple concepts, making interpretability work intractable at the neuron level. Sparse autoencoders (SAEs) solve this by creating 'features,' which are monosemantic dimensions of the autoencoder's latent space. [6] [7]

SAEs are autoencoders which implement the concepts in Sparse Coding. Instead of the latent space being a compressed representation of the input, the autoencoder latent space will be much larger than the input/output dimension. This is done to form an over-complete dictionary of features, the latent space neurons, of the language model (LM). So, the LM will have polysemantic neurons which form the input and output of the SAE, and the latent space will be the interpretable, monosemantic features as the representation. [8] [9]

1.2 Contributions

In this paper, we introduce the variational Sparse Autoencoder (vSAE). We combine VAE and SAE architectures with the hypothesis that probabilistic sampling from Gaussian distributions centered around feature activations will create dispersive pressure, organizing the latent space more coherently and improving interpretability. Empirical findings demonstrate that vSAE underperforms standard SAE across all metrics, with significantly reduced living feature ratios due to KL divergence pressure driving activations toward zero. This computational capacity reduction explains the performance degradation, disproving our initial hypothesis about improved organization and interpretability.

The paper is structured as follows.

- We introduce and define the vSAE architecture, replacing deterministic ReLU gating with stochastic sampling from learned Gaussian posteriors. This includes mathematical formulation of the modified loss function incorporating KL divergence terms and theoretical grounding for why Gaussian sampling should push features apart while organizing similar features together.
- Comprehensive benchmarks using SAE Bench compare vSAE TopK against standard TopK SAE across multiple sparsity levels (k=64, 128, 256, 512) using the Pythia70m LM. We evaluate reconstruction fidelity, dictionary utilization, feature absorption, spurious correlation removal, and targeted probe perturbation to assess competitive performance.
- Local feature analysis through modified SAE Vis examines individual feature interpretability, activation patterns on real dataset samples, synthetic token generation via gradient ascent, and logit effects. We analyze the fraction of living features and compare semantic coherence between vSAE and SAE representations.
- Global latent space analysis using t-SNE dimensionality reduction visualizes feature organization across 20,000 samples. We examine feature clustering patterns, utilization rates, activation magnitudes, and cosine similarity matrices to test whether the hypothesized dispersive pressure materializes in practice and improves spatial organization of learned representations.

1.3 Related Works

Our investigation of variational sparse autoencoders (vSAE) draws from two research fields within the study of autoencoders: variational autoencoders and sparse representation learning. This section surveys the key developments in both areas that inform our approach and experimental design.

1.3.1 VAEs

Variational autoencoders established a principled framework for learning latent variable models through variational inference. [4] The VAE objective combines reconstruction accuracy with a regularization term that encourages the learned posterior to match a chosen prior distribution, typically a standard Gaussian. This approach enables both generative modeling and structured representation learning in a single framework.

The theoretical foundations rest on the Evidence Lower Bound (ELBO), which provides a tractable approximation to the intractable marginal likelihood. The KL divergence term in the ELBO serves as a regularizer that shapes the latent space geometry, encouraging smooth interpolation and meaningful latent structure. [10]

β-VAE introduced a hyperparameter β to weight the KL divergence term, enabling explicit control over the regularization strength. This modification revealed fundamental trade-offs between reconstruction quality and latent disentanglement. [11]

The information bottleneck theory provides a complementary perspective on VAE regularization, framing the KL term as controlling information compression. This theoretical framework helps understand how regularization strength affects the information content preserved in latent representations. [12]

1.3.2 SAEs

early work Using sparse representations to aid in the interpretability of neural networks dates back several decades. Early research involved the application of finding overcomplete sparse dictionary representations by the use of autoencoders on visual data. [13] [14] [15].

The later use of SAEs on LMs were found to produce more interpretable word vectors were found to be successful [16] [17] [18] [19]. Anthropic popularized the notion of features existing in a superposition across the neurons of a neural network [6], which led to the use of sparse autoencoders designed for the use of feature extraction to produce a monosemantic dictionary to represent the activations of a neural network [8] [9] [7].

recent work has seen the augmentation of the SAE to attempt to form more interpretable features while maintaining low reconstruction loss. Different activation and loss functions are introduced to alter which features are present per activation, leading to more structured dictionary sets of features [20] or more control over the noise floor of activations to control sparsity [21]. One choice for sparsity selection is the TopK algorithm [22], which we utilize in our vSAE. TopK takes the top k features active in an activation, then sets the rest to zero to enforce sparsity as a hyperparameter.

The VSEase algorithm [23] applies an adaptive sparsity to the SAE using variational methods, but differs from our approach by introducing the Hadamard product after sampling to gate activations, maintaining sparsity.

2 Mathematical Background

2.1 SAE

The standard Sparse Autoencoder learns a sparse representation of language model activations through deterministic encoding with sparsity regularization. Given input activations $\mathbf{x} \in \mathbb{R}^d$ from a language model's residual stream, the SAE consists of:

Encoder:

$$\mathbf{f} = \text{ReLU}(\mathbf{W}_{\text{enc}}\mathbf{x} + \mathbf{b}_{\text{enc}})$$

where $\mathbf{W}_{\text{enc}} \in \mathbb{R}^{k \times d}$ is the encoder weight matrix, $\mathbf{b}_{\text{enc}} \in \mathbb{R}^k$ is the encoder bias, and $\mathbf{f} \in \mathbb{R}^k$ represents the sparse feature activations with dictionary size k.

TopK Activation:

$$\mathbf{f}_{\text{topk}} = \text{TopK}(\mathbf{f})$$

The TopK operation enforces sparsity by retaining only the K largest activations and setting all others to zero. Formally, if $\mathcal{I}_K$ denotes the indices of the K largest elements in $\mathbf{f}$, then:

$$[\mathbf{f}_{\text{topk}}]_i = \begin{cases} f_i & \text{if } i \in \mathcal{I}_K \\ 0 & \text{otherwise} \end{cases}$$

This hard sparsity constraint ensures exactly K active features per input, providing direct control over the sparsity level without requiring careful tuning of regularization coefficients.

Decoder:

$$\hat{\mathbf{x}} = \mathbf{W}_{\text{dec}}\mathbf{f}_{\text{topk}} + \mathbf{b}_{\text{dec}}$$

where $\mathbf{W}_{\text{dec}} \in \mathbb{R}^{d \times k}$ is the decoder weight matrix and $\mathbf{b}_{\text{dec}} \in \mathbb{R}^d$ is the decoder bias.

Loss Function:

$$\mathcal{L}_{\text{SAE}} = \|\mathbf{x} - \hat{\mathbf{x}}\|_2^2 + \lambda \|\mathbf{f}\|_1$$

where the first term enforces reconstruction fidelity and the second term $\|\mathbf{f}\|_1 = \sum_{i=1}^{k} |f_i|$ is the L1 penalty that enforces sparse activations. The hyperparameter λ controls the sparsity-reconstruction trade-off.

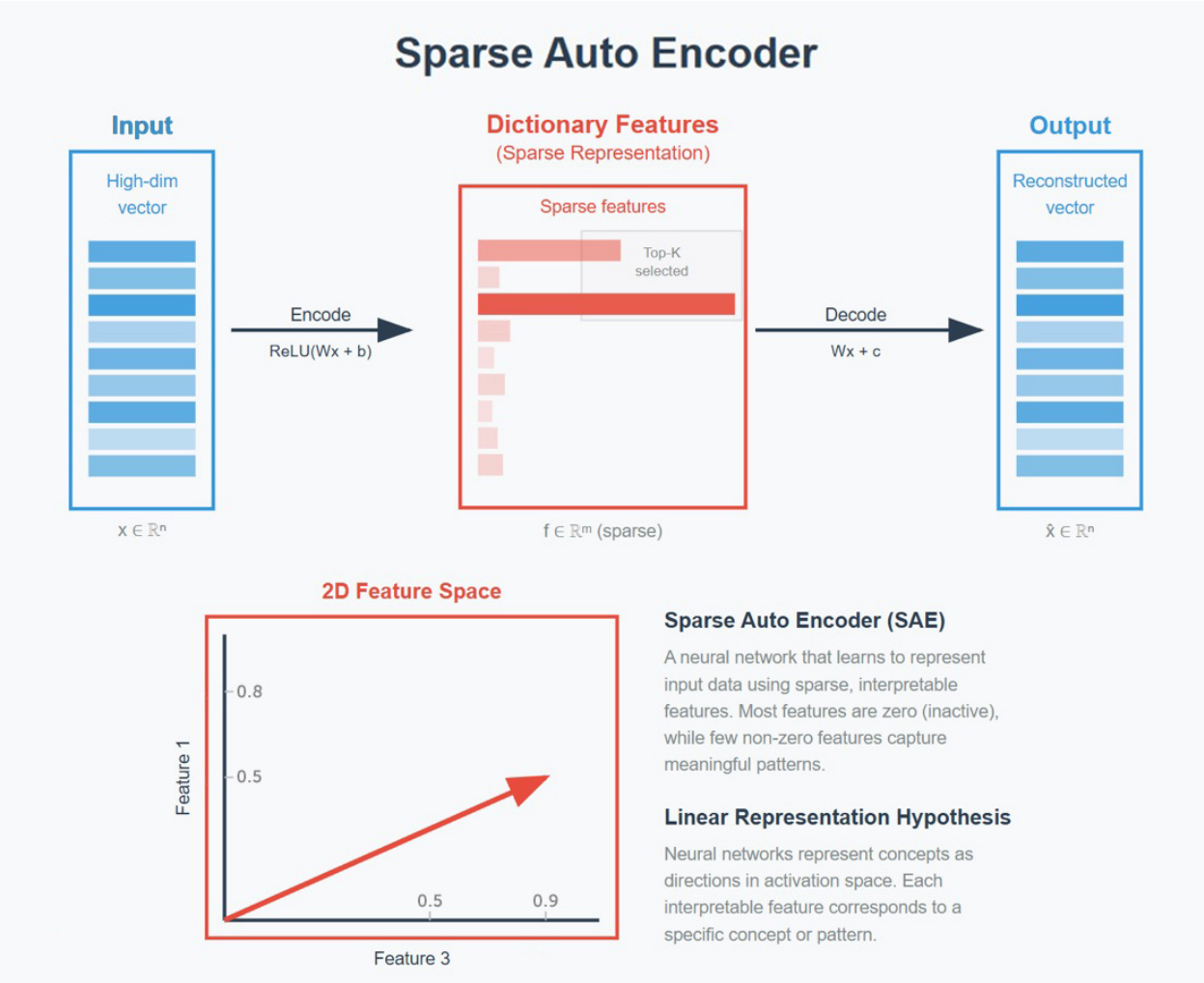

Figure 1: The SAE learns a sparse representation of a language model's (LM) activations through deterministic encoding with sparsity regularization. Here, we use the residual stream of a 1-layer GeLU transformer decoder block as the LM activations (the blue input). The SAE is constructed in a very similar manner to an MLP with one hidden layer.

2.2 vSAE

The variational Sparse Autoencoder incorporates probabilistic modeling by treating feature activations as samples from learned Gaussian distributions rather than deterministic values. This introduces a variational framework with the following components:

Encoder (Mean Estimation):

$$\boldsymbol{\mu} = \mathbf{W}_{\text{enc}}(\mathbf{x} - \mathbf{b}_{\text{dec}}) + \mathbf{b}_{\text{enc}}$$

where $\boldsymbol{\mu} \in \mathbb{R}^k$ represents the mean of the posterior distribution. For simplicity, we fix the variance $\sigma^2 = 1$ for all dimensions.

2.2.1 Reparameterization Trick

A fundamental challenge in training variational autoencoders is that we cannot backpropagate through a sampling operation. During training, we need to sample $\mathbf{z} \sim q(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$ from the learned posterior distribution. However, this sampling operation is stochastic and non-differentiable with respect to the distribution parameters $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$. Without gradients flowing through the sampling step, we cannot update the encoder weights using standard backpropagation.

The reparameterization trick solves this by reformulating the sampling process as a deterministic transformation of a fixed noise distribution:

$$\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\epsilon} \odot \boldsymbol{\sigma}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ is standard Gaussian noise sampled from a distribution independent of the model parameters, $\boldsymbol{\sigma} = \mathbf{1}$ is the fixed standard deviation vector in our case, and $\odot$ denotes element-wise multiplication.

This transformation moves the stochasticity from the latent variable $\mathbf{z}$ to the auxiliary noise variable $\boldsymbol{\epsilon}$. Crucially, $\mathbf{z}$ is now a deterministic function of the learnable parameters $\boldsymbol{\mu}$ and the fixed random noise $\boldsymbol{\epsilon}$. Since the gradient operator doesn't act on $\boldsymbol{\epsilon}$ (it's treated as a constant during backpropagation), we can compute:

$$\frac{\partial \mathbf{z}}{\partial \boldsymbol{\mu}} = \mathbf{I}, \quad \frac{\partial \mathbf{z}}{\partial \boldsymbol{\sigma}} = \boldsymbol{\epsilon}$$

In our vSAE with fixed unit variance ($\boldsymbol{\sigma} = \mathbf{1}$), the reparameterization simplifies to:

$$\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\epsilon}$$

This allows gradients to flow through the decoder and reconstruction loss, enabling end-to-end training via standard backpropagation.

TopK Activation:

$$\mathbf{z}_{\text{topk}} = \text{TopK}(\mathbf{z})$$

Decoder:

$$\hat{\mathbf{x}} = \mathbf{W}_{\text{dec}} \mathbf{z}_{\text{topk}} + \mathbf{b}_{\text{dec}}$$

Loss Function:

$$\mathcal{L}_{\text{vSAE}} = \|\mathbf{x} - \hat{\mathbf{x}}\|_2^2 + \beta \cdot \text{KL}[q(\mathbf{z}|\mathbf{x}) \| p(\mathbf{z})]$$

where: - $q(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$ is the learned posterior distribution - $p(\mathbf{z}) = \mathcal{N}(\mathbf{0}, \mathbf{I})$ is the standard normal prior - β is the KL divergence coefficient

2.2.2 KL Divergence Term

For the isotropic Gaussian case, the KL divergence is defined as:

$$\text{KL}[q(\mathbf{z}|\mathbf{x}) \| p(\mathbf{z})] = \frac{1}{2} \sum_{i=1}^{k} \left(\mu_i^2 + \sigma_i^2 - 1 - \log \sigma_i^2 \right)$$

Since $\sigma_i^2 = 1$ for all i, this further reduces to:

$$\text{KL}[q(\mathbf{z}|\mathbf{x}) \| p(\mathbf{z})] = \frac{1}{2} \sum_{i=1}^{k} \mu_i^2$$

This simplified KL term acts as a regularizer, penalizing the squared L2 norm of the mean vector and encouraging the learned posterior to stay close to the standard normal prior.

2.3 Key Differences

The fundamental distinction lies in sparsity mechanisms:

- Standard SAE: Achieves sparsity through deterministic ReLU gating and explicit L1 regularization - vSAE: Achieves sparsity through stochastic sampling from learned posteriors regularized toward a sparse prior

The vSAE transforms feature representations from deterministic directional vectors into probabilistic "areas" around those directions, hypothetically creating dispersive pressure that organizes features more coherently in the latent space.

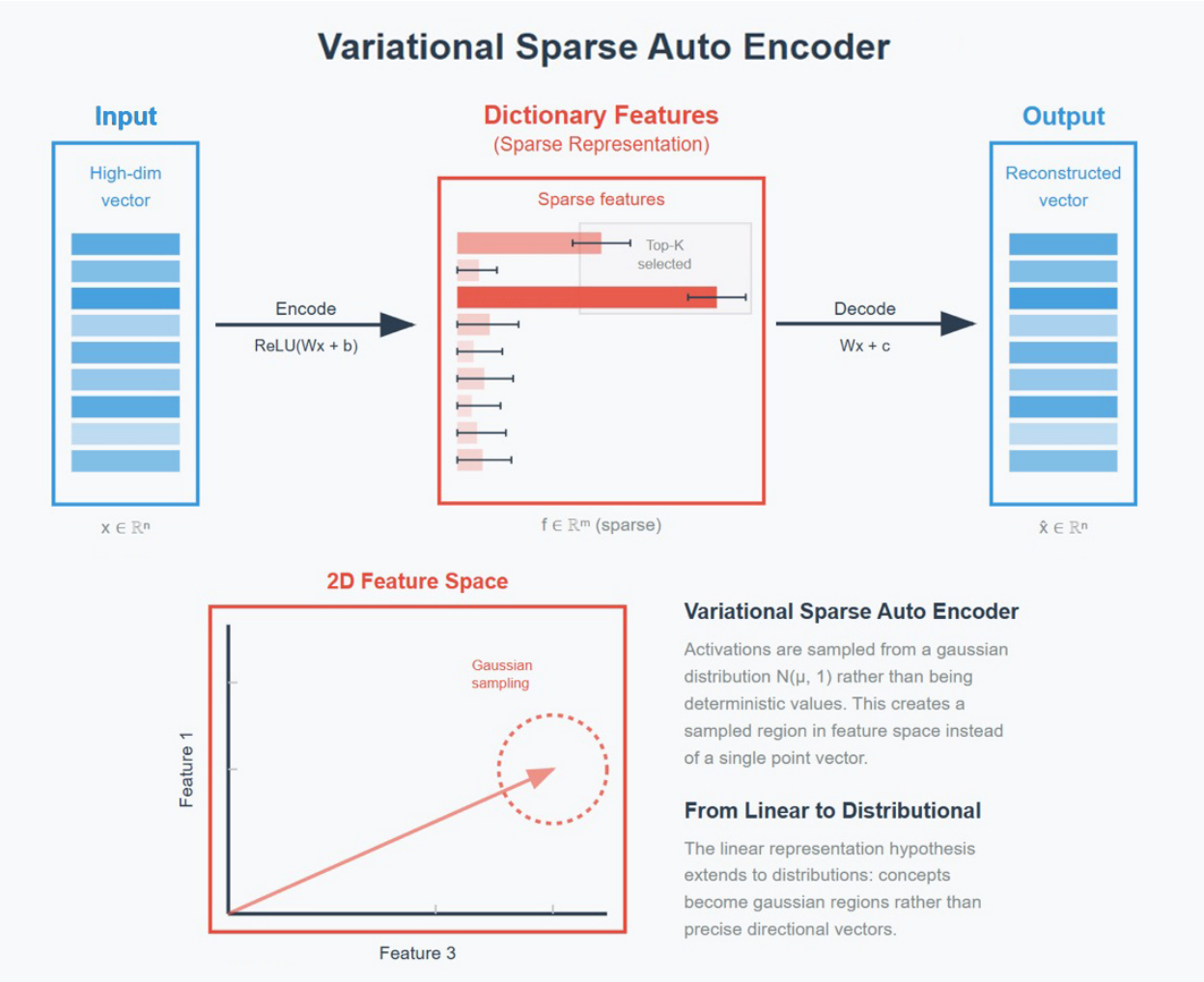

Figure 2: In the vSAE, the activations of the SAE dictionary features are sampled from a gaussian (mean=activation, variance=1), represented by the horizontal bars in the 'dictionary features' box. This changes the feature as a pointer in some direction representing the state of the LM into an area about some direction. We hypothesize that this will cause a 'pressure' for other features to move away from each other, leading to a more organized space of features for the LM state representations.

3 Methods

This study employed a modified version of the Dictionary Learning framework, a widely-used open-source implementation for training sparse autoencoders (SAEs). [24] This framework facilitated benchmark evaluation through SAE Bench and enabled feature visualization via adapted SAE Vis tooling. [25]

The experimental design enables systematic comparison of standard SAE architectures against their variational extensions across multiple model configurations. All evaluations were conducted on the Pythia-70M language model,

specifically targeting layer 0 residual connections (blocks.0.residpost). Dictionary dimensions were set to 4× the MLP width, yielding 2,048 total features.

Training proceeded for 20,000 steps using bfloat16 precision on the TinyStories dataset. Consistent experimental conditions included buffer configurations of 2,500 contexts with 128-token sequences. Performance evaluation employed standard metrics: fraction of variance explained, L0 sparsity, and loss recovery.

Hyperparameter optimization systematically varied learning rates and method-specific parameters, including KL divergence coefficients for variational approaches and TopK values for k-sparse variants, while maintaining architectural consistency across conditions. Variational SAE training incorporated KL annealing schedules to mitigate posterior collapse. All experiments were conducted on NVIDIA RTX 3080 hardware (10GB VRAM).

4 Results

Purely applying the variational methods to the SAE led to poor results: the KL divergence was not a strong enough regularizer to cause sparseness. Therefore, we paired the variational method with the TopK method while running experiments.

We chose to focus on TopK after running extensive studies on SAE architectures (see preliminary analysis here).

We would like to compare the vSAE against its SAE counterpart. We first look at the benchmark results to see how directly competitive the vSAE is. Then we look at the local, specific features which stood out to us. We then look at the global latent analysis of the features using tSNE.

We ultimately found that we lost performance from applying the variational methods on the SAE compared across the other models.

4.1 Benchmarks

SAE Bench [26] is a comprehensive evaluation framework for Sparse Autoencoders that provides standardized benchmarking across several evaluation metrics, including: Feature Absorption, L0/Loss Recovered, Spurious Correlation Removal (SCR), Targeted Probe Perturbation (TPP), and Sparse Probing. We use this tool to compare our vSAE TopK to the standard TopK across different sparsities.

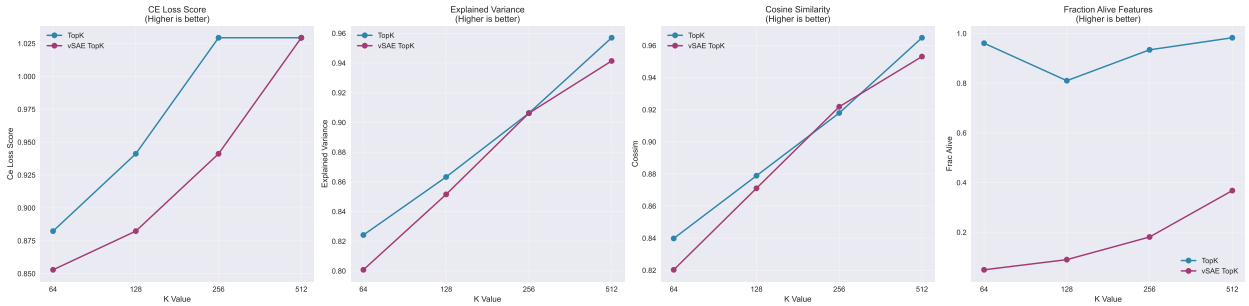

Figure 3: From left to right: CE loss measures how well an SAE reconstructs the original language model's activation to preserve its performance. Explained variance quantifies the proportion of the total variance in the original activations when they are recreated by the SAE's features. High cosine similarity between the original activation and the reconstructed one would indicate good reconstruction quality.

Starting from the left, we can see:

- The CE loss of the SAE and vSAE across multiple sparsities. The SAE consistently outperforms the vSAE, and means that the SAE has representations that can better reconstruct the activations of the language model.
- The explained variance shows that the vSAE is slightly outperformed by the SAE across sparsities, though performance is equal for k=256. In a similar case to the CE loss score, this represents that the SAE has very slightly more faithful representations to the activations of the LM.
- The cosine similarity score is the first time that the vSAE outperforms the SAE, though the gap is small. In general, the LM activation's dot product with the SAE's sparse hidden representations are more aligned than with the vSAE.

- The fourth graph is certainly the most interesting, and shows the key difference in the SAE and the vSAE: the vSAE suffers from many dead features in the benchmark. The massive difference in dead features is likely the reason that we see a slight degradation in performance.

The difference between the two networks tested is the application of the kernel trick and the introduction of the KL loss instead of the traditional l2 loss used in the SAE. As the kernel trick is only a way to allow backpropagation in the network, we surmise that the dead features must be an effect of the KL loss term introduced to the vSAE. The pressure for activations to fit the gaussian footprint causes a quadratic loss pressure onto each feature instead of a linear pressure in the l2 of the SAE. Features unselected by the TopK mechanism still experience the pressure, meaning that many features will be penalized to the point that they stop activating altogether.

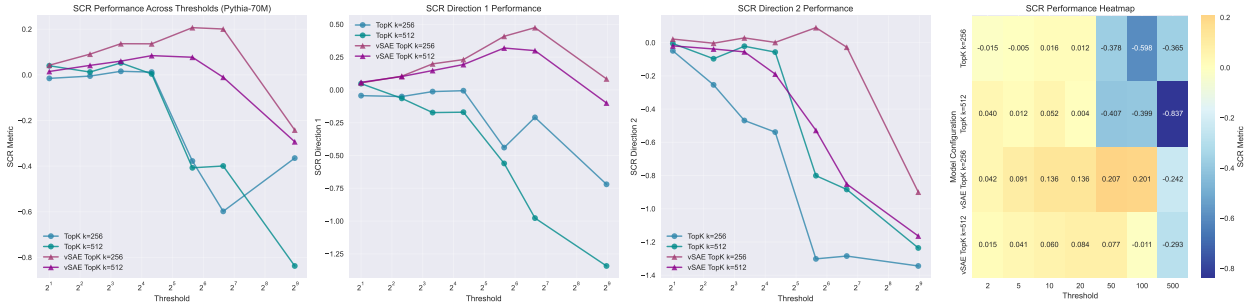

Figure 4: The Spurious Correlation Removal (SCR) is shown for the vSAE and the SAE with k=(256, 512). The SCR is adapted from the SHIFT method [27], a method which improves generalization of an SAE by ablating features that a human judge would find task-irrelevant. The SAEbench paper gives the example of 'gender' and 'profession' as spurious correlations [26]. SCR adapts this SHIFT method by measuring how well a particular SAE allows the removal of these spurious correlations while preserving accuracy. This benchmark measures a type of concept erasure, as a good SAE should have disentangled concepts which can cleanly ablate these spurious concepts. Threshold (x-axis) is correlated with sparsity control: At low thresholds, many latents fire per input, so ablations are less selective. At high thresholds, fewer latents fire — making ablation more targeted. Higher is better, top right is best.

Starting from the left once again, we can see:

- Our vSAE method consistently outperforms the SAE on the SCR benchmark across all sparsities. The SCR scores of vSAE peak higher and remain stable longer before degrading to high thresholds, while the SAE suffers a steep decline near a threshold of 32. Our vSAE achieves its peak around a threshold of 64. This shows that our vSAE is much more robust to ablation than the SAE, and has fewer entangled or diffuse features which are strongly coupled to spurious correlations.

- We see the same relationship in the next graph as before, with the vSAE outperforming the SAE, particularly at thresholds 32-64. The SAE degrades in performance as the threshold moves above 32, indicating that the removal of these spurious signals degrades performance.

- Similar results are shown in the third image. vSAEs maintain positive or mild-negative performance longer, especially K=256, although both models degrade at the threshold of 32 and above. The fourth graph is a numerical representation of the first and introduces no new information.

The performance on the SCR benchmark agrees with our hypothesis: The SAE has spurious correlations that are deeply ingrained in the network, leading to degradation and performance loss when these connections are ablated. Our vSAE, on the other hand, experiences a more independent and less spuriously correlated feature space, so it maintains performance (or even slightly improves) when these spurious connections are removed. The two differences between the SAE and vSAE are the kernel trick and the KL divergence loss term. The kernel trick allows backpropagation while sampling from a distribution, so it is unlikely to be the cause of this change. This leaves the KL divergence term as the cause for the uplift in performance in this benchmark, meaning that the variational method has a more organized latent space capable of eschewing spurious correlation.

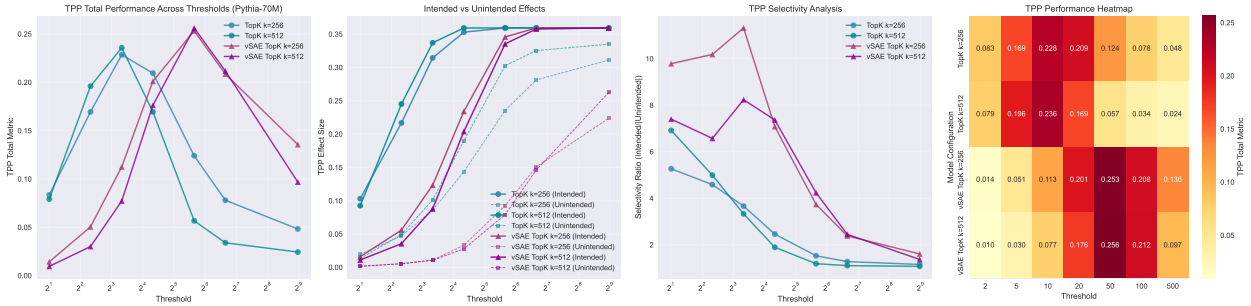

Figure 5: The Targeted Probe Perturbation (TPP) is shown for the vSAE and SAE with k=(256, 512). TPP is a benchmark similar to SCR, but instead of finding spurious correlations (e.g., gender, profession), it instead uses the multi-class setting (i.e., different professions) to measure the entanglement of features, evaluating whether an SAE captures different concepts in distinct sets of latents. TPP rewards interventions that cause a larger performance drop for the target class but minimal change for other classes. Higher is better.

For the final time, starting from the left:

- The SAE peaks around a threshold of 8, while the vSAE has a higher peak near 64. A higher threshold is associated with sparsity, meaning that the SAE peaks while there are relatively more features active with generally lower activation values, and the vSAE peaks with fewer features but with stronger activations of those features. This implies that the vSAE latents are more robust and interpretable, and performs better when fewer features are active.

- This graph introduces unintended effects to pair with the intended effects. We wish to minimize the unintended effects and maximize the intended. The SAE outperforms the vSAE on intended effects, but loses to the vSAE on the unintended effects. This implies that the vSAE is less likely to alter any of its representations if other representations ae ablated, while the SAE will be more severly changed in ways both beneficial and detrimental.

- The high selectivity ratio in the vSAE means that ablating a class will have a small impact on other classes, implying a disentangled representation of the LM activation: the latent space is more dispersed and less entangled than the represenations of the SAE, and implies that the features of the vSAE are more aligned to the classes than the SAE. As the threshold grows, only the stronger latents with high feature activations for the class are still present, showing strong alignment.

The TPP test also agrees with our hypothesis that a vSAE will have a more organized latent space because of the KL divergence in the loss term: the ablation of related concepts in the multiclass setting did not cause severe degradation to the performance of other unintended classes. SCR showed us that spurious concepts did not damage the representations, and even the correlated concepts in this multiclass setting did not damage representations as much as our control, the SAE. According to our benchmarks, the vSAE generally contains more independent representations than that of the SAE.

4.2 Local

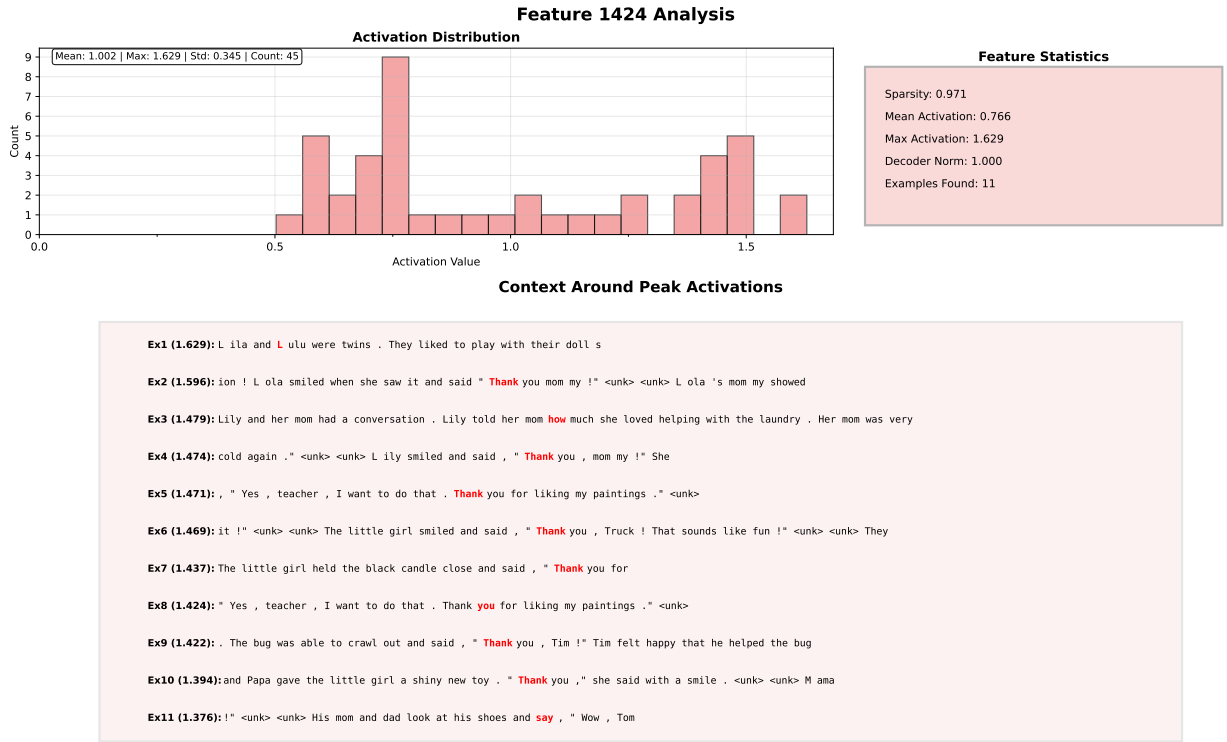

Figure 6: An interpretable feature chosen from the SAE. It appears to represent giving thanks.

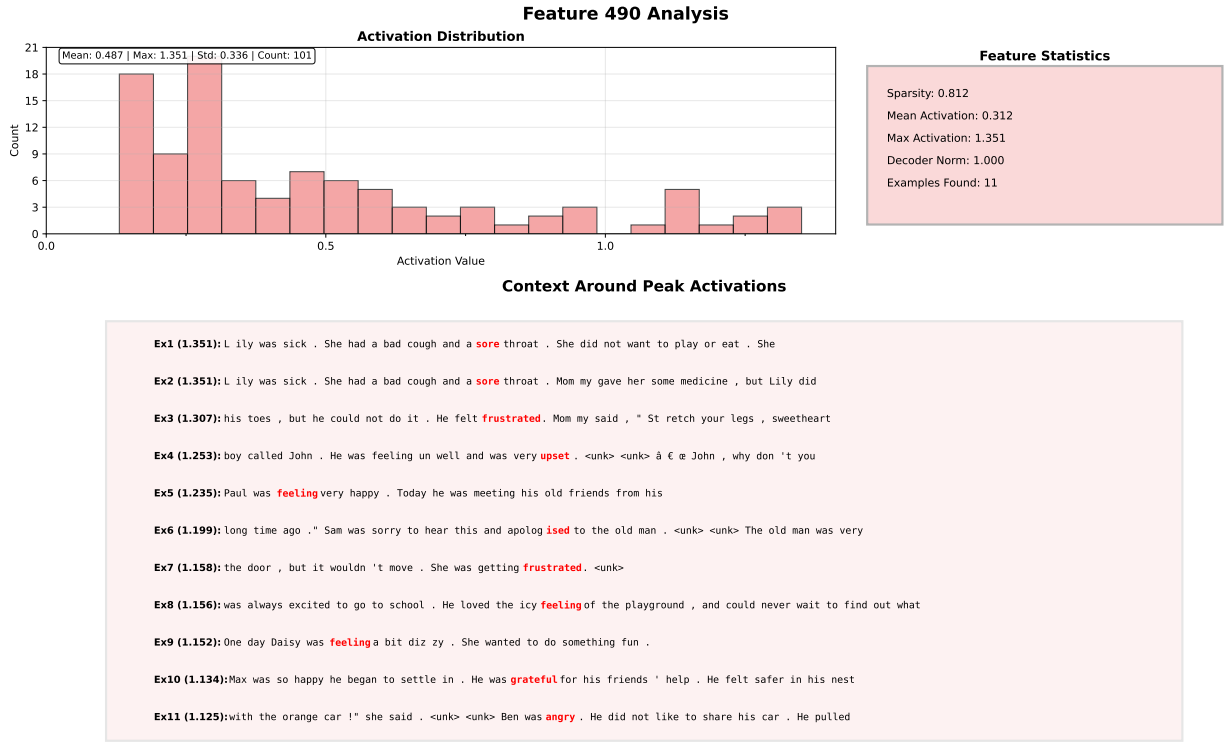

Figure 7: An interpretable feature chosen from the vSAE. It appears to be associated with feelings, especially emotions.

A large difference in the interpretability of either network was not found while investigating features on the TinyStories dataset [28]. However, we found that a far larger percentage of the vSAE features were dead. This is apparent in the following histograms of the maximum activations recorded over features: the vSAE had 82 percent fewer activations recorded compared to the SAE.

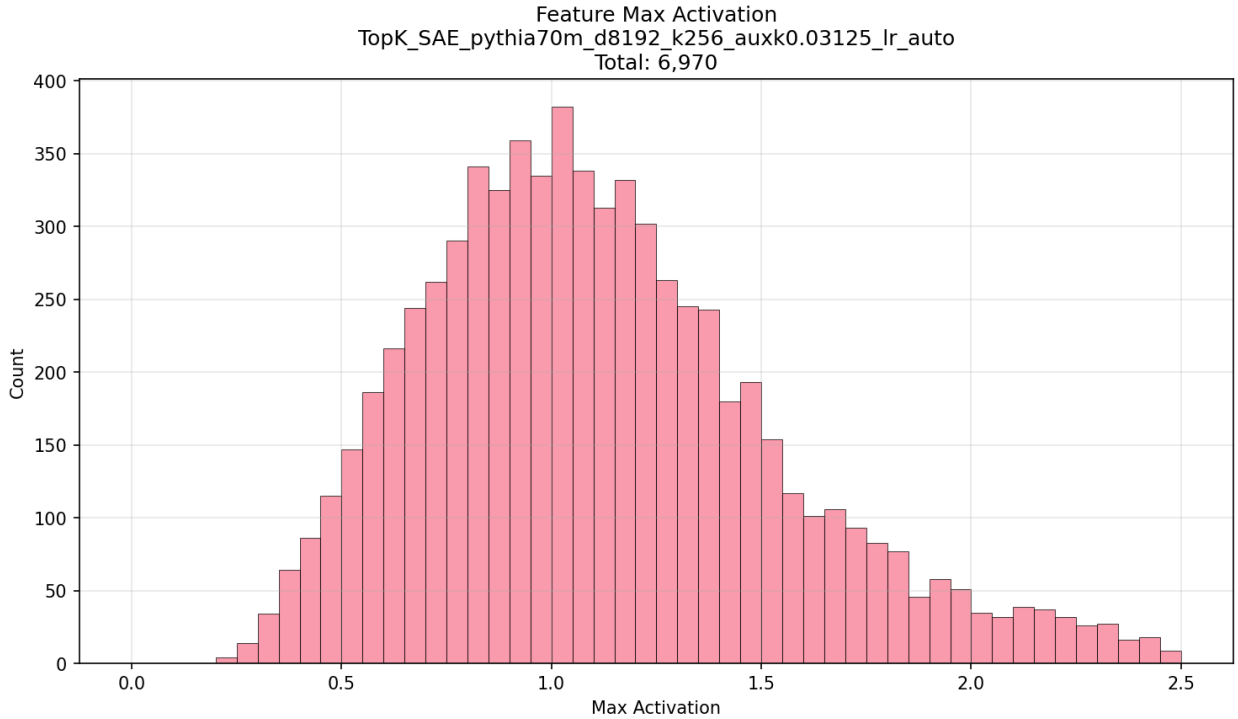

Figure 8: A histogram of the maximum SAE feature activations given over a set of LM activations.

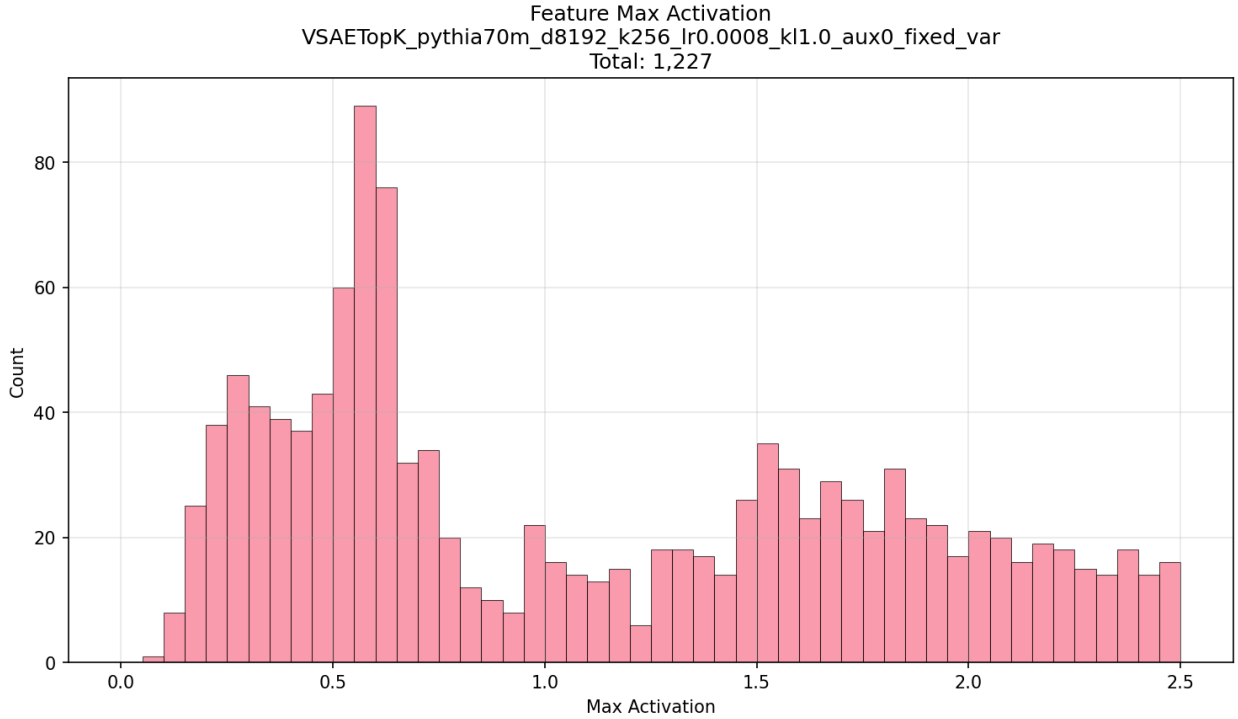

Figure 9: A histogram of the maximum vSAE feature activations given over a set of LM activations. We can see that

there are far fewer activations total, and a different distribution of activations compared to the Gaussian-like structure the SAE formed.

Although only 1,227 features were present in the vSAE histogram compared the SAE's 6,970, we only saw a minor drop in performance. We also saw the vSAE outperform the SAE on the SCR and TPP benchmarks despite having fewer than one sixth of the total available features. The KL loss term has caused both an improvement in the independence of features, but has also deeply limited the computational capacity of the vSAE. The persistence of interpretability of the vSAE features is surprising given the small pool of options available to it, and brings into question the compressibility of model activations.

4.3 Global

To test our hypothesis that variational methods create pressure for features to disperse and organize more coherently, we examine the spatial organization of learned representations through latent analysis. If the Gaussian sampling mechanism successfully pushes features apart as theorized, we expect to observe fewer tightly packed clusters in the vSAE compared to standard SAE representations and a uniform spread. We should see even dispersion in the feature clusters, and a distribution close to uniform in the feature cluster utilization.

Additionally, we analyze whether activation magnitudes distribute evenly across discovered clusters, which would suggest balanced utilization of the representational space rather than concentration in specific regions. Through arbitrary clustering of the learned features, we can quantify these organizational properties and determine whether the probabilistic framework actually delivers the improved spatial organization that motivated our variational approach.

We test across 20,000 LM activation samples using the same LM and dataset from training.

- *Top left (Feature Clusters):* Shows t-SNE dimensionality reduction of the feature space with 10 arbitrary clusters color-coded.

- *Top right (Feature Utilization Rate):* Maps the same t-SNE space but colors points by how frequently each feature activates. Purple indicates rarely-used features (0.1 utilization), while yellow shows heavily-utilized features (0.8). The spatial distribution reveals utilization patterns across the learned representation.

- *Bottom left (Mean Activation, Size=Utilization):* Combines both activation strength (color) and frequency (point size) in the same spatial layout. This reveals whether high-activation features also tend to fire frequently, and whether these properties cluster spatially.

- *Bottom right (Cluster Utilization):* Quantifies utilization rates across the 10 clusters, showing cluster sizes and mean utilization rates.

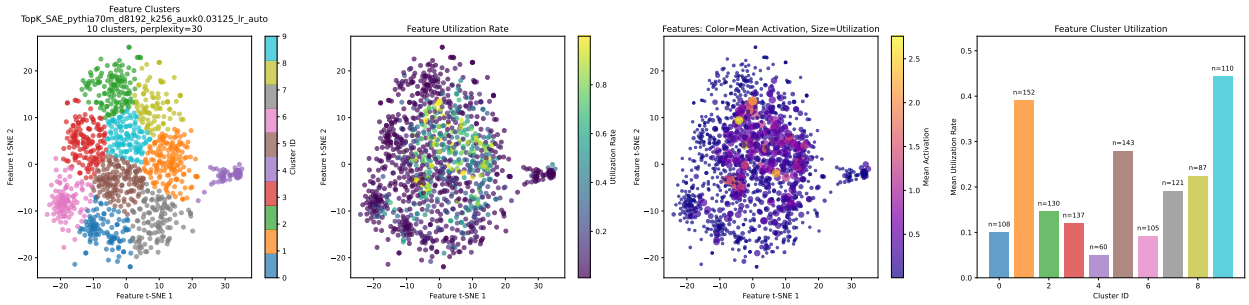

Figure 10: The tSNE visualization of the SAE features. We can see that many features are active (1153 total), and a clear cluster is apparent in the middle right of the feature clusters graph. The feature utilization rate graph shows low utilization clustering in the bottom right, and the middle shows a collection of high-activation features. THis is seem as well in the third graph. The feature cluster utilization is quite varied.

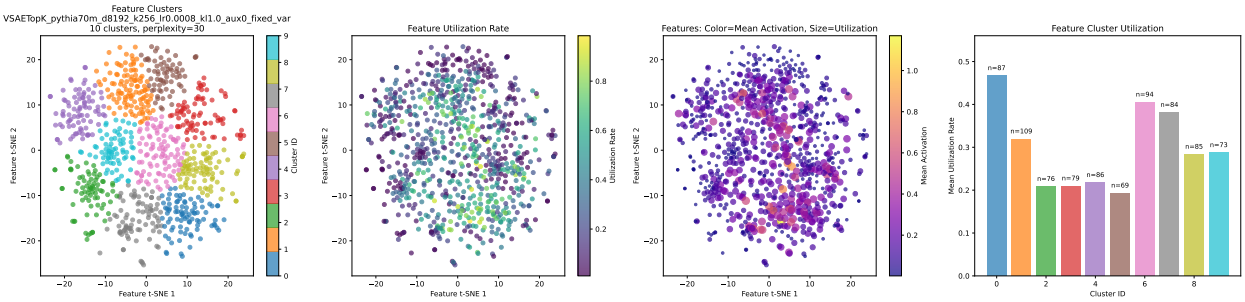

Figure 11: The tSNE visualization of the vSAE features. Less clustering is immediately visible relative to the SAE features, and in the feature utilization rate we can see a greater dispertion of features with higher activations. We can also see a more equal distribution in the feature cluster utilization.

This comparative analysis of latent feature spaces agrees with our central hypothesis that variational methods create pressure for features to disperse and organize more coherently. The vSAE shows clearer dispersion of features than the SAE, meaning that they are less correlated in general. Where the SAE shows predominantly purple regions indicating sparse activation, the vSAE exhibits distributed green and yellow zones throughout the representational space, confirming more balanced utilization rather than concentration in specific spatial regions.

The t-SNE projections show that while both methods produce similar cluster size distributions (vSAE: n=56-151 vs SAE: n=60-152), the vSAE achieves superior spatial dispersion of active features. The Gaussian sampling mechanism successfully pushes features apart in the learned representation, reducing the tight clustering patterns evident in the standard SAE while maintaining greater feature independence and interpretability.

5 Conclusion

The variational Sparse Autoencoder confirms our central hypothesis: probabilistic sampling applied to a TopK sparse autoencoder creates dispersive pressure that organizes features more coherently in latent space. The benchmarks demonstrate this— our vSAE consistently outperforms standard SAE on spurious correlation removal and targeted probe perturbation, showing features with greater independence and less entanglement. The global t-SNE analysis further validates the hypothesis, revealing more uniform spatial distribution and balanced utilization patterns compared to the tightly clustered standard SAE representations. However, the addition of the KL divergence into the loss term also caused the majority of features to die in training, leading to worse performance than our baseline TopK sparse autoencoder in CE loss, explained variance, and cosine similarity tests.

These contradictory results reveal a fundamental trade-off: the KL regularization that improves feature independence simultaneously destroys representational capacity. The variational framework succeeds at its intended goal of creating dispersive pressure but at an unacceptable cost to dictionary utilization.

Disclosure

Claude Sonnet 4 was used for retrieval and discovery in the introduction and works cited sections, as well as to aid and polish in writing. All content retrieved was vetted and analyzed by hand. All content in this paper was written by hand.

References

[1] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning internal representations by error propagation. In *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations*, pages 318–362. MIT Press, Cambridge, MA, 1986.

[2] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[3] Geoffrey E. Hinton and Ruslan R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[4] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2013.

[5] Anika Shrivastava, Renu Rameshan, and Samar Agnihotri. Latent space characterization of autoencoder variants, 2024.

[6] Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg, and Christopher Olah. Toy models of superposition. *Transformer Circuits Thread*, 2022.

[7] 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.

[8] Lee Sharkey, Dan Braun, and Beren Millidge. Taking features out of superposition with sparse autoencoders, 2022. AI Alignment Forum.

[9] Hoagy Cunningham, John Hewitt, Giovanni Gulli, Robert Gotschall, and Robert Huben. Sparse autoencoders find highly interpretable features in language models. *arXiv preprint arXiv:2309.08600*, 2023.

[10] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models, 2014.

[11] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-VAE: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*, 2017.

[12] Naftali Tishby and Noga Zaslavsky. Deep learning and the information bottleneck principle, 2015.

[13] Honglak Lee, Alexis Battle, Rajat Raina, and Andrew Ng. Efficient sparse coding algorithms. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems*, volume 19. MIT Press, 2006.

[14] Marc' aurelio Ranzato, Christopher Poultney, Sumit Chopra, and Yann Cun. Efficient learning of sparse representations with an energy-based model. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems*, volume 19. MIT Press, 2006.

[15] Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y. Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, page 609–616, New York, NY, USA, 2009. Association for Computing Machinery.

[16] Manaal Faruqui, Yulia Tsvetkov, Dani Yogatama, Chris Dyer, and Noah Smith. Sparse overcomplete word vector representations, 2015.

[17] Sanjeev Arora, Yuanzhi Li, Yingyu Liang, Tengyu Ma, and Andrej Risteski. Linear algebraic structure of word senses, with applications to polysemy, 2018.

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

[19] Zeyu Yun, Yubei Chen, Bruno A Olshausen, and Yann LeCun. Transformer visualization via dictionary learning: contextualized embedding as a linear superposition of transformer factors, 2023.

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

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

[22] 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, 2024.

[23] Yin Lu, Xuening Zhu, Tong He, and David Wipf. Sparse autoencoders, again?, 2025.

[24] Samuel Marks, Adam Karvonen, and Aaron Mueller. dictionary learning. `https://github.com/saprmarks/dictionary_learning`, 2024.

[25] Callum McDougall. SAE Visualizer. `https://github.com/callummcdougall/sae_vis`, 2024.

[26] Adam Karvonen, Can Rager, Johnny Lin, Curt Tigges, Joseph Bloom, David Chanin, Yeu-Tong Lau, Eoin Farrell, Callum McDougall, Kola Ayonrinde, Demian Till, Matthew Wearden, Arthur Conmy, Samuel Marks, and Neel Nanda. Saebench: A comprehensive benchmark for sparse autoencoders in language model interpretability, 2025.

[27] Samuel Marks, Can Rager, Eric J. Michaud, Yonatan Belinkov, David Bau, and Aaron Mueller. Sparse feature circuits: Discovering and editing interpretable causal graphs in language models, 2025.

[28] Ronen Eldan and Yuanzhi Li. Tinystories: How small can language models be and still speak coherent english?, 2023.