FaithfulSAE: Towards Capturing Faithful Features with Sparse Autoencoders without External Dataset Dependencies

Anonymous ACL submission

Abstract

Sparse Autoencoders (SAEs) have emerged as a promising solution for decomposing large language model representations into interpretable features. However, Paulo and Belrose (2025) have highlighted instability across different initialization seeds, and Heap et al. (2025) have pointed out that SAEs may not capture model-internal features. These problems likely stem from training SAEs on external datasets—either collected from the Web or generated by another model—which may contain out-of-distribution (OOD) data beyond the model's generalisation capabilities. This can result in hallucinated SAE features, which we term "Fake Features", that misrepresent the model's internal activations. To address these issues, we propose FaithfulSAE, a method that trains SAEs on the model's own synthetic dataset. Using FaithfulSAEs, we demonstrate that training SAEs on less-OOD instruction datasets results in SAEs being more stable across seeds. Notably, FaithfulSAEs outperform SAEs trained on web-based datasets in the SAE probing task and exhibit a lower Fake Feature Ratio in 5 out of 7 models. Overall, our approach eliminates the dependency on external datasets, advancing interpretability by better capturing model-internal features while highlighting the often neglected importance of SAE training datasets.

1 Introduction

Sparse Autoencoders (SAEs), an architecture introduced by Faruqui et al., 2015, have demonstrated the ability to transform Large Language Model (LLM) representations into interpretable features without supervision (Huben et al., 2023). SAE latent dimensions can be trained to reconstruct activations while incurring a sparsity penalty, ideally resulting in a sparse mapping of human-interpretable features. This approach enables decomposition of latent representations into interpretable features by reconstructing transformer hidden states (Gao et al., 2024) or MLP activations (Bricken et al., 2023b).

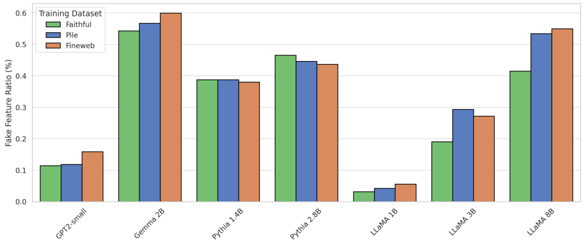

Figure 1: Fake Feature Ratio for SAEs trained on Faithful dataset and Web-based datasets (lower is better). Detailed values can be found in Table 7.

Despite the demonstrated utility of SAE features, several concerns persist: SAEs can yield very different feature sets depending on the initialization seed (Paulo and Belrose, 2025), SAEs can exhibit highly activated latents which reduce interpretability (Stolfo et al., 2025; Smith et al., 2025), and when trained on random or out-of-distribution data, SAEs often capture dataset artifacts rather than genuine model-internal patterns (Heap et al., 2025; Bricken et al., 2023b). Such spurious dimensions can be viewed as hallucinated SAE features (henceforth, "Fake Features") that misrepresent the model's true activations.

This work investigates SAE reliability issues, hypothesizing that this unreliability stems from out-of-distribution (OOD) datasets in LLMs (Yang et al., 2023; Liu et al., 2024), which are defined as datasets not generalized in LLMs, either absent from pretraining or too complex for the model's capabilities. To compare the effects of OOD datasets, a Faithful dataset is generated, self-generated synthetic dataset by the LLM, to more accurately reflect LLM-intrinsic features and capabilities. Faithful SAEs are trained on this dataset and their "faithfulness" is evaluated by measuring reconstruction performance with Cross Entropy (CE), L2 loss, and Explained Variance metrics, while using feature matching techniques (Balagansky et al., 2025; Laptev et al., 2025; Paulo and Belrose, 2025) to assess stability across different seeds.

Based on our experiments, SAEs trained on OOD datasets yield feature sets sensitive to seed differences and lack robustness across different datasets. First, SAEs were trained on instruction dataset using non-instruction-tuned Pythia (Biderman et al., 2023) models, representing naturally OOD data. Second, Faithful datasets were compared with potentially OOD Web datasets with different model architectures. Results showed visible differences in stability across seeds between instruction datasets and Faithful Datasets, while such differences were less pronounced against Web datasets. Additionally, SAEs trained on Web datasets showed unstable faithfulness across datasets with the above metrics, when compared to FaithfulSAEs.

2 Background

2.1 Mechanistic Interpretability

Mechanistic Interpretability encompasses approaches that reverse-engineer neural networks through examination of their underlying mechanisms and intermediate representations (Olah et al., 2020; Elhage et al., 2021). Researchers systematically analyse multidimensional latent representations, uncovering phenomena such as layer pattern features (Olah et al., 2017; Carter et al., 2019) and neuron-level features (Goh et al., 2021; Schubert et al., 2021) within vision models. The development of the attention mechanism (Vaswani et al., 2017) and Transformer architecture has intensified research into understanding the emergent capabilities of these models (Wei et al., 2022b).

2.2 Superposition Hypothesis

Within neural networks' representational space, the superposition of word embeddings (Arora et al., 2018) has provided substantial evidence for superposition phenomena. Through studies with toy models, Elhage et al. 2022 elaborated on how the superposition hypothesis emerges via Phase Change in feature dimensionality, establishing connections to compressed sensing (Donoho, 2006; Bora et al., 2017). This hypothesis suggests that polysemanticity emerges as a consequence of neural networks optimizing their representational capacity. Research has demonstrated that transformer activations contain significant superposition (Gurnee et al., 2023), suggesting these models encode information as linear combinations of sparse, independent features.

2.3 Sparse Autoencoders

Sparse Autoencoders (Huben et al., 2023; Bricken et al., 2023b) address the Superposition Hypothesis in Transformers by disentangling representational patterns through sparse dictionary learning (Olshausen and Field, 1997; Elad, 2010) for the underlying features. These models are structured as overcomplete autoencoders, featuring hidden layers with greater dimensionality than their inputs, while incorporating sparsity constraints through L_1 regularisation or explicit TopK mechanisms (Gao et al., 2024). Their architectural diversity encompasses various activation functions including ReLU (Dunefsky et al., 2024), JumpReLU (Rajamanoharan et al., 2025), TopK (Gao et al., 2024), BatchTopK (Bussmann et al., 2024), alongside different regularisation approaches and decoding mechanisms.

2.4 SAE Feature

The SAE features refer to the simplest factorization of hidden activations, which are expected to be human-interpretable latent activations for certain contexts (Bricken et al., 2023a). However, sparsity and reconstruction are competing objectives; minimizing loss may occur without preserving conceptual (Leask et al., 2025) coherence, as sparsity loss randomly suppresses features, which may cause low reproducibility in SAEs. Moreover, SAEs trained with different seeds or hyperparameters often converge to different sets of features (Paulo and Belrose, 2025). This instability challenges the assumption that SAEs reliably uncover a unique, model-intrinsic feature dictionary.

2.5 SAE Weight

The SAE reconstructs the activations through the following process:

$$x_{\text{feature}} = \sigma(x_{\text{hidden}} \cdot W_{\text{enc}} + b_{\text{enc}}) \quad (1)$$

$$\hat{x}_{\text{hidden}} = x_{\text{feature}} \cdot W_{\text{dec}} + b_{\text{dec}} \quad (2)$$

where σ is the activation function.

The encoder weight matrix multiplication can be represented in two forms that yield the same result:

$$x_{\text{feature}} = \sigma\left(\sum_{i=1}^{A}(a_i \cdot w_{i,\cdot}^{\text{enc}}) + b_{\text{enc}}\right) \tag{3}$$

$$x_{\text{feature}} = \sigma\left(\bigoplus_{j=1}^{D}(x_{\text{hidden}} \cdot w_{\cdot,j}^{\text{enc}} + b_j^{\text{enc}})\right) \tag{4}$$

where A is the activation size and D is the dictionary size and $\bigoplus$ denotes group concatenation.

- $w_{i,\cdot}^{\text{enc}}$: Each row of the encoder matrix represents the coefficients for linearly disentangling a hidden representation's superposition.
- $w_{\cdot,j}^{\text{enc}}$: Each column of the encoder matrix represents the coefficients for linearly composing a hidden representation from monosemantic features.
- $w_{i,j}^{\text{enc}}$: The specific weight at index (i, j) indicates how much the jth feature contributes to the superposition at the ith hidden representation.

The decoder weight matrix multiplication can also be represented in two forms that yield the same result:

$$\hat{x}_{\text{hidden}} = \sum_{j=1}^{D}(d_j \cdot w_{j,\cdot}^{\text{dec}} + b_j^{\text{dec}}) \tag{5}$$

$$\hat{x}_{\text{hidden}} = \bigoplus_{i=1}^{A}(x_{\text{feature}} \cdot w_{\cdot,i}^{\text{dec}}) + b_{\text{dec}} \tag{6}$$

- $w_{j,\cdot}^{\text{dec}}$: Each row of the decoder matrix shows dictionary features in hidden activations, a Feature Direction (Templeton et al., 2024) that capture the direction of the feature in the hidden space.
- $w_{\cdot,i}^{\text{dec}}$: Each column of the decoder matrix shows how each monosemantic dictionary feature contributes to the reconstructed hidden superposition.
- $w_{j,i}^{\text{dec}}$: The specific weight at index (j, i) specifies how feature j is composited to reconstruct hidden representation i.

This formulation underscores the critical role of the encoder and decoder weights in disentangling features and accurately reconstructing hidden activations.

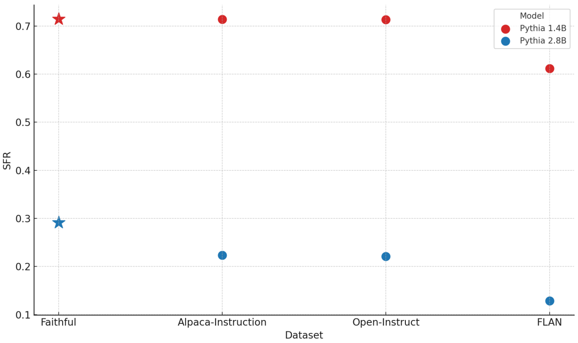

Figure 2: Shared Feature Ratio (SFR) comparison between Faithful Dataset and Instruction Dataset trained SAEs. Detailed values for each run are listed in Table 2.

3 Methods

3.1 Faithful Dataset Generation

To develop Faithful SAEs that accurately reflect the capabilities of LLMs, the training dataset should closely align with the model's inherent distribution. The model's generative distribution was captured through unconditional sampling, providing only the Beginning-of-Sequence (BOS) token as the input prompt. This is referred to as the Faithful Dataset, as it directly corresponds to the model's natural next-token prediction distribution.

3.2 Faithful SAE Training

Using the generated Faithful Dataset, the Top-K SAEs (Gao et al., 2024) were trained. To demonstrate the faithfulness of the trained models, two Faithful SAEs were trained with the same configuration but different seeds. For comparison, SAEs with the same seeds were also trained using not only the SAE dataset but also various other datasets.

3.3 Evaluation Metrics

Faithfulness was evaluated by examining individual learned features in the SAE latent space across different seeds, with specific metrics as follows. To quantify the faithfulness of SAEs, several complementary metrics were employed. The primary metrics include Shared Feature Ratio, Cross-Entropy (CE) difference, L2 reconstruction error, and Explained Variance.

3.4 Feature Matching

To understand how different training conditions affect the learned representations within SAEs, features discovered by different SAEs are compared using Feature Matching (Balagansky et al.,

Model	Total Tokens	Vocab Size	All Token Coverage (%)	First Token Coverage (%)	KL (Model → Dataset)
GPT-2 Small	110,718,964	50,257	99.80	21.49	0.2631
Pythia 1.4B	99,999,541	50,254	99.31	5.43	1.0498
Pythia 2.8B	103,204,690	50,254	99.04	3.14	1.1198
Pythia 6.9B	57,580,971	50,254	99.41	13.38	0.2893
Gemma 2B	121,006,576	256,000	93.44	0.40	2.2392
LLaMA 3.2-1B	110,070,117	128,000	95.78	8.27	0.1521
LLaMA 3.2-3B	110,395,870	128,000	96.09	9.18	0.1909
LLaMA 3.1-8B	180,268,487	128,000	98.04	10.31	0.1054

Table 1: Token statistics across models in the Faithful dataset. KL (Model → Dataset) represents the forward KL divergence between generated dataset's first token distribution and BOS prediction distribution.

2025; Laptev et al., 2025; Paulo and Belrose, 2025). A common approach, inspired by Maximum Marginal Cosine Similarity (MMCS) (Sharkey et al., 2022), computes the cosine similarity between feature vectors using their corresponding decoder weight vectors, where $w_j = w_{j,\cdot}^{dec}$.

$$m_j = \max_{w'_k \in W_2} \frac{w_j \cdot w'_k}{\|w_j\| \, \|w'_k\|}$$

Following Paulo and Belrose (2025), the Hungarian matching algorithm (Kuhn, 1955) was used to find an optimal one-to-one correspondence between feature sets. We compute the similarity matrix $S \in \mathbf{R}^{d \times d}$ between all features of two SAEs:

$$S_{j,k} = \frac{w_{j,\cdot}^{dec} \cdot w_{k,\cdot}^{dec'}}{\|w_{j,\cdot}^{dec}\| \, \|w_{k,\cdot}^{dec'}\|}$$

After applying the Hungarian algorithm to find the optimal assignment that maximizes the total similarity, each feature is classified based on a threshold τ_s into 'shared' or 'orphan' features, terminology introduced by Paulo and Belrose (2025):

$$\text{Feature Type}(d_j) = \begin{cases} \text{shared} & \text{if } S_{j,k} \geq \tau_s, \\ \text{orphan} & \text{if } S_{j,k} < \tau_s. \end{cases}$$

This approach ensures that each feature from one SAE is matched with at most one feature from the other SAE, providing a measure of feature set similarity.

Using this methodology, the Shared Feature Ratio is defined as the proportion of shared features relative to the total number of features in an SAE:

$$SFR = \frac{|\{d_j \in D \mid S_{j,k} \geq \tau_s\}|}{|D|}$$

where D is the complete dictionary of features in the SAE, and $|\cdot|$ denotes the cardinality of a set.

3.5 Fake Feature Ratio

Frequently activating features have been identified as problematic in SAE literature (Stolfo et al., 2025; Smith et al., 2025), often leading to poor interpretability. "Fake Feature" is defined as a feature that activate on randomly generated token sequences (OOD inputs). A feature is considered fake if it frequently activates on more than a certain threshold τ_f of OOD samples. The Fake Feature Ratio (FFR) is defined as:

$$\text{FFR} = \frac{|\{i \in D : \text{activation frequency}(i) > \tau_f\}|}{|D|}$$

where D is the total feature dictionary. Lower FFR indicates better feature quality.

3.6 SAE Probing

To evaluate downstream task performance of SAE, three approaches are compared on classification tasks: original model activations (Baseline), sparse feature activations (SAE), and reconstructed activations (Reconstruction). Logistic regression probes are trained for each representation type and accuracy and F1 scores are measured across SST-2, CoLA, AG News, and Yelp Polarity datasets. A faithful SAE should show minimal performance drop between baseline and SAE/reconstruction approaches.

4 Experiments

We used SFR with threshold τ_s as 0.7 between SAEs trained with different random seeds. For the FFR threshold, we followed Smith et al. (2025) and

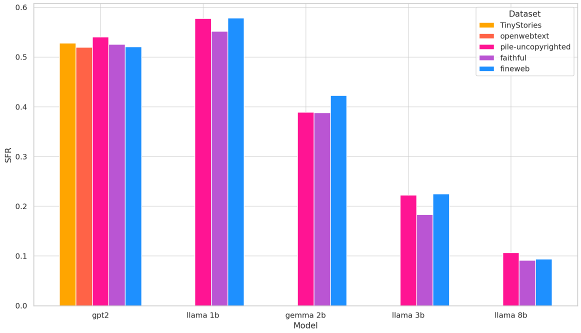

Figure 3: Shared Feature Ratio by model and dataset. SAE training hyperparameters are listed in Appendix A, and complete results appear in Table 4.

set $\tau_f = 0.1$. For each experiment, we trained multiple SAEs using two different initialization seeds while keeping all other hyperparameters constant. For all datasets except LLaMA 8B, we used 100M tokens for training. For LLaMA 8B, we used 150M tokens to ensure convergence. FFR measurement was measured by generating 1M tokens and averaged across all different seed SAEs for a reliable measure.

4.1 Instruction Dataset Comparison

The training dataset used during pre-training must be publicly available. For example, models like LLaMA (Team, 2024b) do not disclose their training data. The research leveraged the fact that pre-trained models have internalised the distribution of their training data and rely on this distribution for inference. Therefore, the pre-trained model was treated as a proxy for its training distribution and used to generate synthetic data. The open-source Pythia (Biderman et al., 2023) model was employed, for which the training dataset is publicly available.

For the Out-of-Distribution (OOD) datasets, Instruction Tuning (Wei et al., 2022a) datasets were used: FLAN (Longpre et al., 2023), OpenInstruct (Wang et al., 2023), and Alpaca dataset (Taori et al., 2023). Selecting an uncensored dataset was crucial for constructing a valid OOD benchmark. This decision was based on the fact that commonly used datasets for training SAEs contain data scraped from the same sources. Additionally, models with different parameter scales were compared: Pythia 1.4B and Pythia 2.8B, to study the impact of model size on SAE faithfulness.

4.2 Web-based Dataset Comparison

For cross-architecture comparison against Web-based dataset and Faithful dataset, the Top-K SAE model (Gao et al., 2024) was utilized. To evaluate a diverse range of architectures and examine scaling effects, five models were employed: GPT-2 Small (Radford et al., 2019), LLaMA 3.2 1B, LLaMA 3.2 3B, LLaMA 3.1 8B (Team, 2024b), and Gemma 2B (Team, 2024a). SAEs were trained on three distinct datasets—The Pile (Gao et al., 2021), FineWeb (Penedo et al., 2024), and our Faithful Dataset—for each model architecture, with hyperparameters specified in Table 5. After training SAEs across different datasets and architectures using two initialization seeds, the SFR metric was compared when only the seed was altered to assess model stability.

4.3 SAE Faithfulness Metrics

The objective is to determine whether training SAEs on the generated Faithful dataset produces more faithful sparse representations of model activations. It is argued that a more faithful SAE should adapt more flexibly to the model when encoding and decoding activations, maintaining the essential information flow through the model. To quantify this faithfulness, Cross-Entropy (CE) difference, L2 reconstruction error, and Explained Variance were used as proxy metrics, comparing trained SAEs to measure their impact on the underlying model. This evaluation was conducted using SAEs trained on The Pile, FineWeb, and the Faithful Dataset, and extended the test suite to include not only these three datasets but also OpenWebText (Gokaslan and Cohen, 2019) and TinyStories (Li and Eldan, 2024) for comprehensive assessment.

4.4 SAE Probing

For our SAE Probing experiments, four diverse classification datasets were selected: SST-2 (Socher et al., 2013), CoLA (Warstadt et al., 2019), AG News and Yelp Polarity (Zhang et al., 2015). For each dataset, reconstructed activations were used as input for logistic regression classifier. Activations were aggregated by mean pooling on every token in the sequence. The classifiers were trained on each representation type and accuracy score was measured, using a maximum of 100,000 samples for training. The accuracy scores were averaged across all seed SAEs to obtain more reliable data.

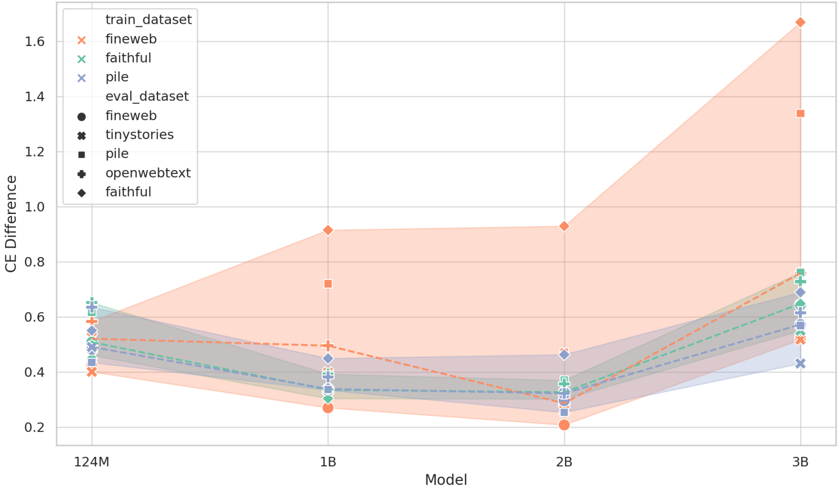

Figure 4: Cross-Entropy difference between SAEs trained on different datasets. Colors represent training datasets: orange for FineWeb, gray for Pile-Uncopyrighted, and green for Faithful dataset. Point shapes indicate evaluation datasets: circles for FineWeb, squares for The Pile, X markers for TinyStories, crosses for OpenWebText, and diamonds for Faithful dataset. You can find the detailed metrics in Appendix B.

5 Results

5.1 Impact of OOD Levels on SAE Stability Across Datasets

As shown in Table 2, FaithfulSAEs, trained on a synthetic dataset, exhibit greater stability across seeds compared to SAEs trained on mixed or instruction-based datasets. These results support our hypothesis that higher OOD levels reduce SFR. Notably, layer 16 demonstrates higher stability than layer 8, likely due to SAEs capturing more complex features in deeper layers.

Dataset	Pythia 1.4B	Pythia 2.8B
Faithful	**0.7145**	**0.2911**
Alpaca-Instruction	0.7138	0.2231
Open-Instruct	0.7134	0.2210
FLAN	0.6113	0.1283

Table 2: Shared Feature Ratio for Pythia 1.4B and 2.8B model. AI denotes Alpaca-Instruction for compactness.

5.2 SFR on Cross-Model Synthetic Datasets

From Table 3, we observe that SFR is consistently higher when the target model is the same as the source model (e.g., training SAEs on a Pythia 2.8B model with a synthetic dataset from a 2.8B model), and lower when the source and target models are different. This suggests that SAE training on its own synthetic dataset is more stable even within the same model family trained on the same dataset with different scaling. This indicates that SFR differences stem from out-of-distribution effects, and a smaller model's dataset is not necessarily easier to learn stable feature sets from. The results are consistent with our hypothesis: more OOD input leads to lower SAE stability across seeds (lower SFR), while less OOD leads to more consistent SAE training (higher SFR).

Target Model	Source Model	SFR
Pythia 2.8b	Pythia 2.8b	**0.2911**
Pythia 2.8B	Pythia 1.4B	0.2288
Pythia 1.4B	Pythia 1.4B	**0.7145**
Pythia 1.4B	Pythia 2.8B	0.6887

Table 3: Shared Feature Ratio on Pythia models. FaithfulSAEs were trained on target models with synthetic datasets generated from source models.

5.3 Performance on Web-based Datasets

The Faithful dataset did not demonstrate higher SFR compared to web-based datasets as shown in Figure 3; rather, it showed lower SFR across most models. As evident in Table 4, the Faithful dataset exhibited lower SFR than FineWeb or The Pile for all models.

We concluded that this issue arises because web-based datasets are sufficiently diverse to encompass model coverage, and out-of-distribution data beyond the scope of the Faithful dataset does not negatively impact the robustness of SAEs.

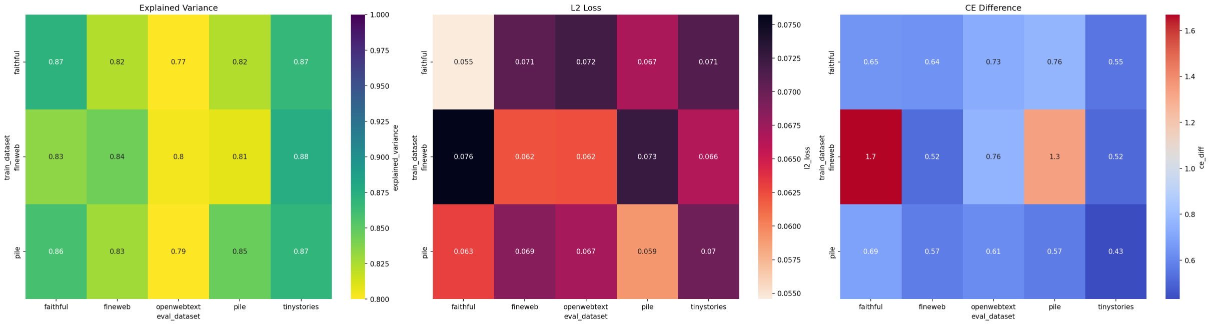

Figure 5: Faithful SAE representation for LLaMa 8B. This figure shows the SAE's reconstruction of the LLaMa 8B hidden state and its faithfulness across datasets.

Model	Pile	Faithful	FineWeb
GPT-2	**0.5405**	0.5258	0.5209
LLaMA 1B	0.5778	0.5517	**0.5789**
Gemma 2B	0.3889	0.3881	**0.4229**
LLaMA 3B	0.2222	0.1835	**0.2248**
LLaMA 8B	**0.1066**	0.0914	0.0936

Table 4: Shared Feature Ratio across models and datasets. It compares SAEs trained with identical settings but different seeds. The models listed were used for SAE activation extraction, and the datasets on the right were used for training them.

By observing that GPT2 relatively showed similar SFR with other Web-based datasets, while the larger models such as Gemma and LLaMA consistently showed lower SFR. This is because the pretraining datasets of Gemma and LLaMA already contain Web-based data generalization, which means they are not OOD datasets. To address this limitation, generating larger Faithful datasets would better cover the full range of model capabilities, which we analyze in more detail in Subsection 5.4 by comparing SAE faithfulness.

5.4 Faithfulness of Faithful Dataset

As shown in Table 1, KL divergence values stay below 2 except for Gemma 2B, demonstrating effective mode covering via Forward KL. The table confirms >90% Unique Tokens Used in All Positions, indicating adequate model distribution capture. However, first token distribution lacks vocabulary breadth, possibly explaining why Figure 3 shows FaithfulSAEs underperforming Web-based SAEs. Alternative approaches include starting with a flat distribution instead of BOS tokens or increasing the sampling temperature.

In Appendix C, we verify the proper generation of the dataset by confirming that the distribution of top tokens follows the predicted distribution of BOS tokens. However, due to limited sampling in the dataset, it does not cover all token distributions from the BOS prediction, which follow a logarithmic decrease.

5.5 Faithfulness of FaithfulSAE

To determine whether training SAEs on the generated Faithful dataset produces more faithful SAEs, we evaluated model fidelity during activation encoding and decoding processes with trained SAEs as presented in Table 5. We measured Cross-Entropy difference, L2, and Explained Variance metrics across five datasets. The full results are available in Appendix B, while the results for LLaMa 8B are shown in Figure 5.

Although FineWeb SAE showed higher SFR than Faithful SAE, it demonstrated significantly higher CE difference and overall lower generalized performance on faithfulness metrics. SAEs trained on The Pile achieved higher SFR, while faithfulness metrics were similar as shown in Appendix B. SAEs trained exclusively on the Faithful Dataset demonstrated more stable performance across multiple evaluation datasets compared to FineWeb.

5.6 SAE Probing

Notably in Figure 6, FaithfulSAE demonstrates overall better performance compared to the other Web-based trained SAEs. FaithfulSAE achieved superior performance in 12 out of 18 cases across six models and three classification tasks. While performance varied by task, FaithfulSAE consistently outperformed alternatives on the CoLA dataset across all model configurations. Despite showing lower SFR compared to Web-based datasets, the higher downstream task performance of FaithfulSAE suggests it more accurately reflects the model's hidden state with less reconstruction noise.

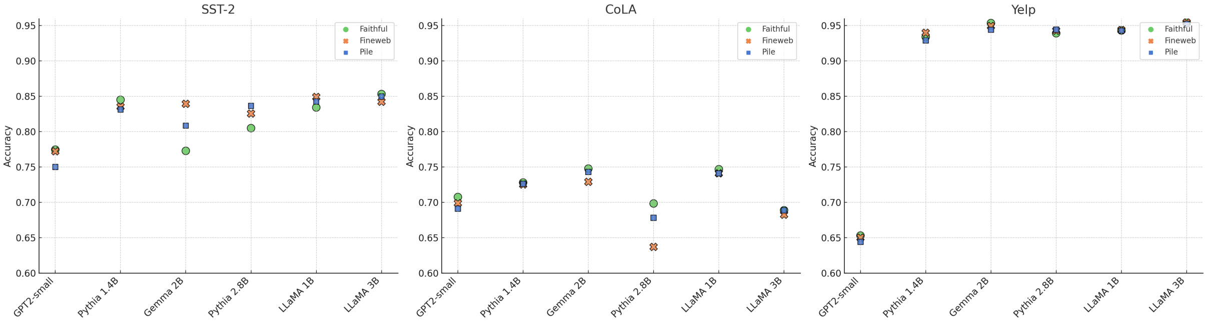

Figure 6: SAE Probing performance comparison between FaithfulSAE and Web-based SAEs with different types of LLM architectures. Detailed values can be found in Table 6.

5.7 Fake Feature

While FaithfulSAE generally shows lower SFR compared to web-based datasets, it demonstrates better performance in terms of FFR (lower), suggesting potential benefits for interpretability with the Faithful Dataset. Among the 7 models tested, 5 models showed lower FFR with FaithfulSAE, with the exception of the Pythia model family. This is likely because the Pythia model, as mentioned above, was trained exclusively on The Pile dataset, which closely overlaps with the web-based FineWeb and The Pile datasets used for comparison. We also observed that within the same model family, larger models showed higher FFR with FaithfulSAE, indicating that interpretability becomes more challenging as model size increases.

6 Conclusion

Out-of-distribution datasets that exceed a model's pretraining distribution or capabilities hinder SAEs from reliably identifying consistent feature sets across different initialization seeds. To mitigate this, we proposed Faithful SAE—trained on the model's own synthetic dataset—to ensure that training remains strictly within the model's inherent capabilities. Our experiments showed that FaithfulSAEs yield higher SFR than those trained on instruction-tuned datasets and outperform SAEs trained on Web-based datasets in the SAE proving task. While FaithfulSAEs obtain lower FFR than web-based dataset trained SAEs leading to improved potential interpretability, they also offer a key advantage: encapsulation.

7 Limitations

While Faithful Datasets improve feature consistency for non-instruction-tuned models, our experiment lacked evaluation on instruction-tuned or reasoning models. Our evaluation of Shared Feature Ratio may not fully reflect the complexity of high-dimensional feature spaces, and we did not assess the interpretability of individual features. Specifically, Shared Feature Ratio was higher compared to instruction datasets, but lower compared to web-based datasets. Additionally, we need to verify whether Faithful SAE provides interpretable explanations for individual features through case studies. Although we defined the Fake Feature Ratio and confirmed lower values, we did not remove these features or assess their interpretability further.

8 Future Work

This work shows that our approach can reduce Fake Features and improve probing performance. An important direction for future research is exploring improved dataset generation and training strategies that could completely outperform Web-based methods. Such progress would further validate the promise of training interpretability models using only the model itself, without reliance on external data. This dataset independence could be particularly advantageous for interpretability in domain-specific generative models where data is scarce. For example, the FaithfulSAE approach could be adopted for interpretability of models in biology or robotics where data production costs are high.

Another priority is to evaluate whether Faithful SAEs provide meaningful and interpretable explanations for individual features through detailed case studies. For example, we hypothesize that pruning Fake Features from a Faithful SAE may yield a representation close to the Simplest Factorization (Bricken et al., 2023a), aligning with the principle of Minimal Description Length (Ayonrinde et al., 2024). Confirming this connection remains an open and exciting avenue for future investigation.

References

Sanjeev Arora, Yuanzhi Li, Yingyu Liang, Tengyu Ma, and Andrej Risteski. 2018. Linear algebraic structure of word senses, with applications to polysemy. *Transactions of the Association for Computational Linguistics*, 6:483–495.

Kola Ayonrinde, Michael T. Pearce, and Lee Sharkey. 2024. Interpretability as compression: Reconsidering sae explanations of neural activations with mdl-saes. *Preprint*, arXiv:2410.11179.

Nikita Balagansky, Ian Maksimov, and Daniil Gavrilov. 2025. Mechanistic permutability: Match features across layers. In *The Thirteenth International Conference on Learning Representations*.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, Usvsn Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar Van Der Wal. 2023. Pythia: A suite for analyzing large language models across training and scaling. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 2397–2430. PMLR.

Ashish Bora, Ajil Jalal, Eric Price, and Alexandros G Dimakis. 2017. Compressed sensing using generative models. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, volume 70 of *Proceedings of Machine Learning Research*, pages 537–546, Sydney, Australia. PMLR.

Trenton Bricken, Joshua Batson, Adly Templeton, Adam Jermyn, Tom Henighan, and Chris Olah. 2023a. Features as the simplest factorization. Part of the May 2023 Circuits Updates by the Anthropic interpretability team.

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, and 6 others. 2023b. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*.

Bart Bussmann, Patrick Leask, and Neel Nanda. 2024. Batchtopk sparse autoencoders. In *NeurIPS 2024 Workshop on Scientific Methods for Understanding Deep Learning*.

Shan Carter, Zan Armstrong, Ludwig Schubert, Ian Johnson, and Chris Olah. 2019. Activation atlas. *Distill*.

D.L. Donoho. 2006. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306.

Jacob Dunefsky, Philippe Chlenski, and Neel Nanda. 2024. Transcoders find interpretable LLM feature circuits. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Michael Elad. 2010. *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing*, 1 edition. Springer New York.

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. 2022. Toy models of superposition. *Transformer Circuits Thread*.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, and 6 others. 2021. A mathematical framework for transformer circuits. *Transformer Circuits Thread*.

Manaal Faruqui, Yulia Tsvetkov, Dani Yogatama, Chris Dyer, and Noah A. Smith. 2015. Sparse overcomplete word vector representations. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1491–1500, Beijing, China. Association for Computational Linguistics.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. 2021. The pile: An 800gb dataset of diverse text for language modeling. *CoRR*, abs/2101.00027.

Leo Gao, Tom Dupre la Tour, Henk Tillman, Gabriel Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan Leike, and Jeffrey Wu. 2024. Scaling and evaluating sparse autoencoders. In *The Thirteenth International Conference on Learning Representations*.

Gabriel Goh, Nick Cammarata †, Chelsea Voss †, Shan Carter, Michael Petrov, Ludwig Schubert, Alec Radford, and Chris Olah. 2021. Multimodal neurons in artificial neural networks. *Distill*.

Aaron Gokaslan and Vanya Cohen. 2019. Openwebtext corpus. `https://skylion007.github.io/OpenWebTextCorpus/`.

Wes Gurnee, Neel Nanda, Matthew Pauly, Katherine Harvey, Dmitrii Troitskii, and Dimitris Bertsimas. 2023. Finding neurons in a haystack: Case studies with sparse probing. *Transactions on Machine Learning Research*.

Thomas Heap, Tim Lawson, Lucy Farnik, and Laurence Aitchison. 2025. Sparse autoencoders can interpret randomly initialized transformers. *Preprint*, arXiv:2501.17727.

Robert Huben, Hoagy Cunningham, Logan Riggs Smith, Aidan Ewart, and Lee Sharkey. 2023. Sparse autoencoders find highly interpretable features in language models. In *The Twelfth International Conference on Learning Representations*.

Harold W. Kuhn. 1955. The hungarian method for the assignment problem. *Naval Research Logistics (NRL)*, 52.

Daniil Laptev, Nikita Balagansky, Yaroslav Aksenov, and Daniil Gavrilov. 2025. Analyze feature flow to enhance interpretation and steering in language models. *Preprint*, arXiv:2502.03032.

Patrick Leask, Bart Bussmann, Michael T Pearce, Joseph Isaac Bloom, Curt Tigges, Noura Al Moubayed, Lee Sharkey, and Neel Nanda. 2025. Sparse autoencoders do not find canonical units of analysis. In *The Thirteenth International Conference on Learning Representations*.

Yuanzhi Li and Ronen Eldan. 2024. Tinystories: How small can language models be and still speak coherent english.

Bo Liu, Li-Ming Zhan, Zexin Lu, Yujie Feng, Lei Xue, and Xiao-Ming Wu. 2024. How good are LLMs at out-of-distribution detection? In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 8211–8222, Torino, Italia. ELRA and ICCL.

Shayne Longpre, Le Hou, Tu Vu, Albert Webson, Hyung Won Chung, Yi Tay, Denny Zhou, Quoc V Le, Barret Zoph, Jason Wei, and Adam Roberts. 2023. The flan collection: Designing data and methods for effective instruction tuning. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 22631–22648. PMLR.

Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. 2020. Zoom in: An introduction to circuits. *Distill*. Https://distill.pub/2020/circuits/zoom-in.

Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. 2017. Feature visualization. *Distill*.

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

Gonçalo Paulo and Nora Belrose. 2025. Sparse autoencoders trained on the same data learn different features. *Preprint*, arXiv:2501.16615.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro Von Werra, and Thomas Wolf. 2024. The fineweb datasets: Decanting the web for the finest text data at scale. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI*. Accessed: 2024-11-15.

Senthooran Rajamanoharan, Tom Lieberum, Nicolas Sonnerat, Arthur Conmy, Vikrant Varma, Janos Kramar, and Neel Nanda. 2025. Jumping ahead: Improving reconstruction fidelity with jumpreLU sparse autoencoders.

Ludwig Schubert, Chelsea Voss, Nick Cammarata, Gabriel Goh, and Chris Olah. 2021. High-low frequency detectors. *Distill*.

Lee Sharkey, Dan Braun, and Beren Millidge. 2022. Interim research report: Taking features out of superposition with sparse autoencoders. AI Alignment Forum, posted December 13, 2022.

Lewis Smith, Senthooran Rajamanoharan, Arthur Conmy, Callum McDougall, Tom Lieberum, János Kramár, Rohin Shah, and Neel Nanda. 2025. Negative results for saes on downstream tasks and deprioritising sae research. `https://www.lesswrong.com/posts/4uXCAJNuPKtKBsi28/sae-progress-update-2-draft`. DeepMind Mechanistic Interpretability Team Progress Update #2.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Alessandro Stolfo, Ben Peng Wu, and Mrinmaya Sachan. 2025. Antipodal pairing and mechanistic signals in dense SAE latents. In *ICLR 2025 Workshop on Building Trust in Language Models and Applications*.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Alpaca: A Strong, Replicable Instruction-Following Model.

Gemma Team. 2024a. Gemma 2: Improving open language models at a practical size. *Preprint*, arXiv:2408.00118.

Llama Team. 2024b. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, Craig Citro, Emmanuel Ameisen, Andy Jones, Hoagy Cunningham, Nicholas L Turner, Callum McDougall, Monte MacDiarmid, C. Daniel Freeman, Theodore R. Sumers, Edward Rees, Joshua Batson, Adam Jermyn, and 3 others. 2024. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. *Transformer Circuits Thread*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Yizhong Wang, Hamish Ivison, Pradeep Dasigi, Jack Hessel, Tushar Khot, Khyathi Chandu, David Wadden, Kelsey MacMillan, Noah A. Smith, Iz Beltagy, and Hannaneh Hajishirzi. 2023. How far can camels go? exploring the state of instruction tuning on open resources. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. 2022a. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022b. Emergent abilities of large language models. *Transactions on Machine Learning Research*. Survey Certification.

Linyi Yang, Yaoxian Song, Xuan Ren, Chenyang Lyu, Yidong Wang, Jingming Zhuo, Lingqiao Liu, Jindong Wang, Jennifer Foster, and Yue Zhang. 2023. Out-of-distribution generalization in natural language processing: Past, present, and future. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4533–4559, Singapore. Association for Computational Linguistics.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Appendix

A SAE Training

For the SAE training, the learning rates and TopK values roughly followed the scaling laws proposed by Gao et al. (2024). 100 M tokens were used for all datasets except for LLaMA 8B, where 150 M tokens were used to ensure convergence. All SAE training was conducted using an NVIDIA RTX 3090ti 24GB. Additionally, to obtain a sufficiently complex feature set when training a single layer, we used the target layer at the 3/4 position except Gemma2 2B model. For the uncensored instruction dataset, we utilized FLAN[1], Open-Instruct [2], and Alpaca dataset [3] in our experiments.

Model	Layer	DictSize	TopK	LR	Seed	Dataset	Sequence Length
GPT2-small	8	12288	48	0.0002	42,49	Faithful-gpt2-small	128
GPT2-small	8	12288	48	0.0002	42,49	Pile-uncopyrighted	128
GPT2-small	8	12288	48	0.0002	42,49	FineWeb	128
GPT2-small	8	12288	48	0.0002	42,49	OpenWebText	128
GPT2-small	8	12288	48	0.0002	42,49	TinyStories	128
Llama-3.2-1B	12	14336	48	0.0002	42,49	Faithful-llama3.2-1b	512
Llama-3.2-1B	12	14336	48	0.0002	42,49	Pile-uncopyrighted	512
Llama-3.2-1B	12	14336	48	0.0002	42,49	Fineweb	512
Gemma-2-2b	20	18432	64	0.0003	42,49	Faithful-gemma2-2b	1024
Gemma-2-2b	20	18432	64	0.0003	42,49	Pile-uncopyrighted	1024
Gemma-2-2b	20	18432	64	0.0003	42,49	Fineweb	1024
Llama-3.2-3B	21	18432	64	0.0001	42,49	Faithful-llama3.2-3b	512
Llama-3.2-3B	21	18432	64	0.0001	42,49	Pile-uncopyrighted	512
Llama-3.2-3B	21	18432	64	0.0001	42,49	Fineweb	512
Llama-3.1-8B	24	16384	80	6e-05	42,49	Faithful-llama3.1-8b	512
Llama-3.1-8B	24	16384	80	6e-05	42,49	Pile-uncopyrighted	512
Llama-3.1-8B	24	16384	80	6e-05	42,49	Fineweb	512
Pythia-1.4B	18	14336	48	0.0002	42,49	Faithful-pythia-1.4b	512
Pythia-1.4B	18	14336	48	0.0002	42,49	Faithful-pythia-2.8b	512
Pythia-1.4B	18	14336	48	0.0002	42,49	Open-Instruct	512
Pythia-1.4B	18	14336	48	0.0002	42,49	Alpaca-Instruction	512
Pythia-1.4B	18	14336	48	0.0002	42,49	FLAN	512
Pythia-2.8B	24	15360	64	0.0001	42,49	Faithful-pythia-1.4b	512
Pythia-2.8B	24	15360	64	0.0001	42,49	Faithful-pythia-2.8b	512
Pythia-2.8B	24	15360	64	0.0001	42,49	Open-Instruct	512
Pythia-2.8B	24	15360	64	0.0001	42,49	Alpaca-instruction	512
Pythia-2.8B	24	15360	64	0.0001	42,49	FLAN	512

Table 5: SAE training hyperparameters for each model and dataset. The configuration includes the model name, layer index, dictionary size, top-k sparsity, learning rate, random seed, training dataset, and sequence/token dimensions. (a) and (b) are shorthand tags used for table compactness.

[1] https://huggingface.co/datasets/Open-Orca/FLAN

[2] https://huggingface.co/datasets/xzuyn/open-instruct-uncensored-alpaca

[3] https://huggingface.co/datasets/aifeifei798/merged_uncensored_alpaca

B Faithful SAEs

Below figures show how each SAE trained on different datasets generalizes its reconstruction capability on other datasets, reflecting its faithfulness. It compares the Explained Variance, L2 loss, and CE difference across datasets when the LLM's hidden state is replaced by the SAE's reconstructed activation trained on a specific dataset. The X-axis represents the evaluation dataset, and the Y-axis indicates the SAE's training dataset. All results are based on SAE models trained with seed 42.

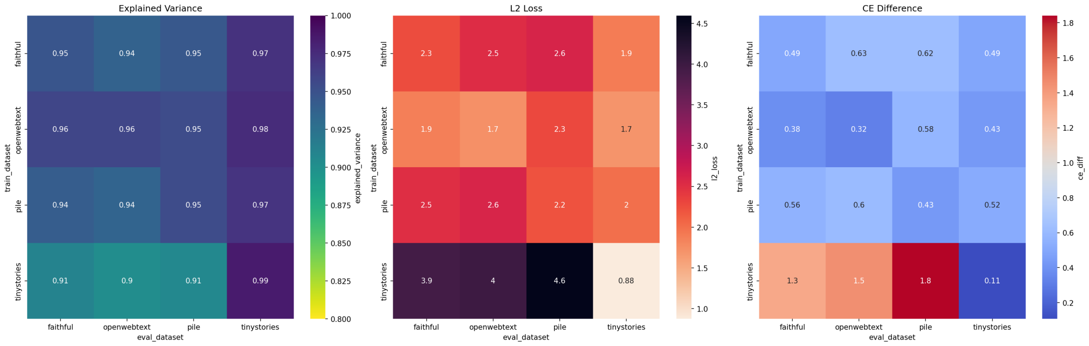

Figure 7: Faithful SAE representation for GPT-2. This figure visualizes the SAE model's ability to reconstruct GPT-2's hidden state.

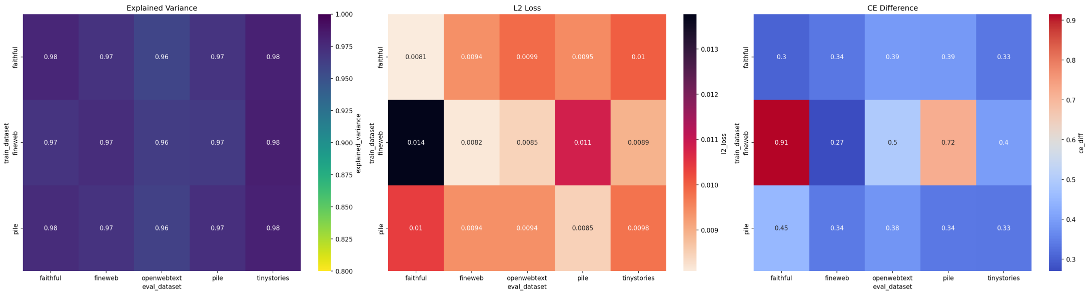

Figure 8: Faithful SAE representation for LLaMA 1B. This figure demonstrates the SAE's performance in reconstructing the hidden state of LLaMA 1B.

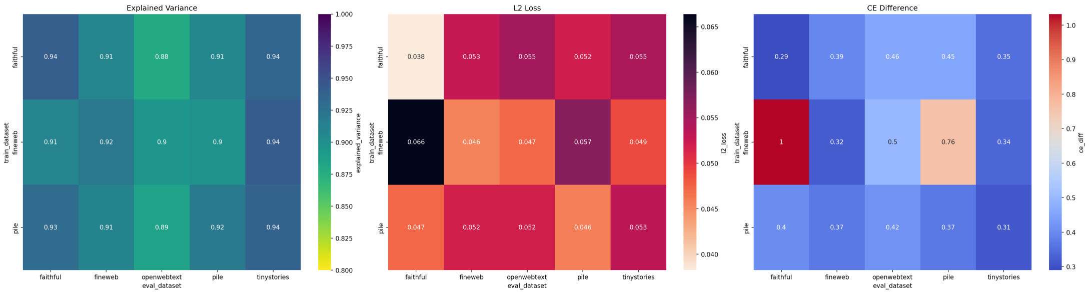

Figure 9: Faithful SAE representation for LLaMA 3B. This figure highlights the SAE's reconstruction quality for the LLaMA 3B model's hidden state.

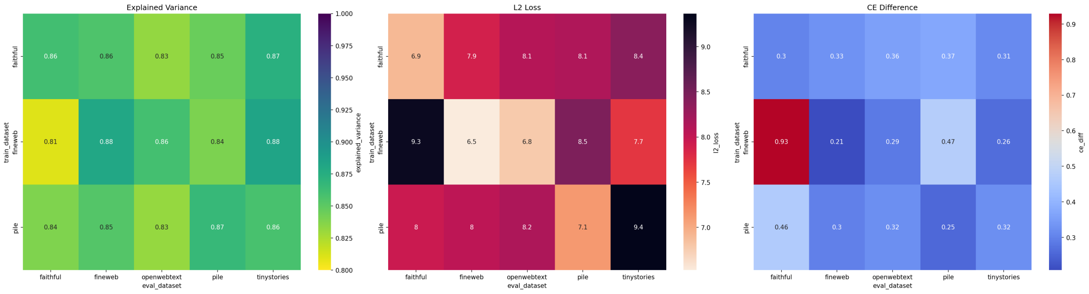

Figure 10: Faithful SAE representation for Gemma 2B. This figure shows the SAE's reconstruction of the Gemma 2B hidden state and its faithfulness across datasets.

C Faithful Dataset

Below figures compare the model's BOS token's next token distribution and the real frequency distribution of the first token from our generated Faithful dataset. The left two figures represent the model's distribution, and the right two figures represent the dataset's token frequency distribution. The upper two figures only select the top 10 tokens, which show almost identical shapes to the original model. However, the bottom two graphs show that the frequency distribution does not cover the whole token distribution, as the probability decreases exponentially for the first generation. By comparing the coverage and token statistics, we verified that the Faithful dataset reflects the original model's capability well. Additionally, the Pythia 6.9B model was only used for dataset generation to check the token distribution matching the model's BOS token and was not used for training.

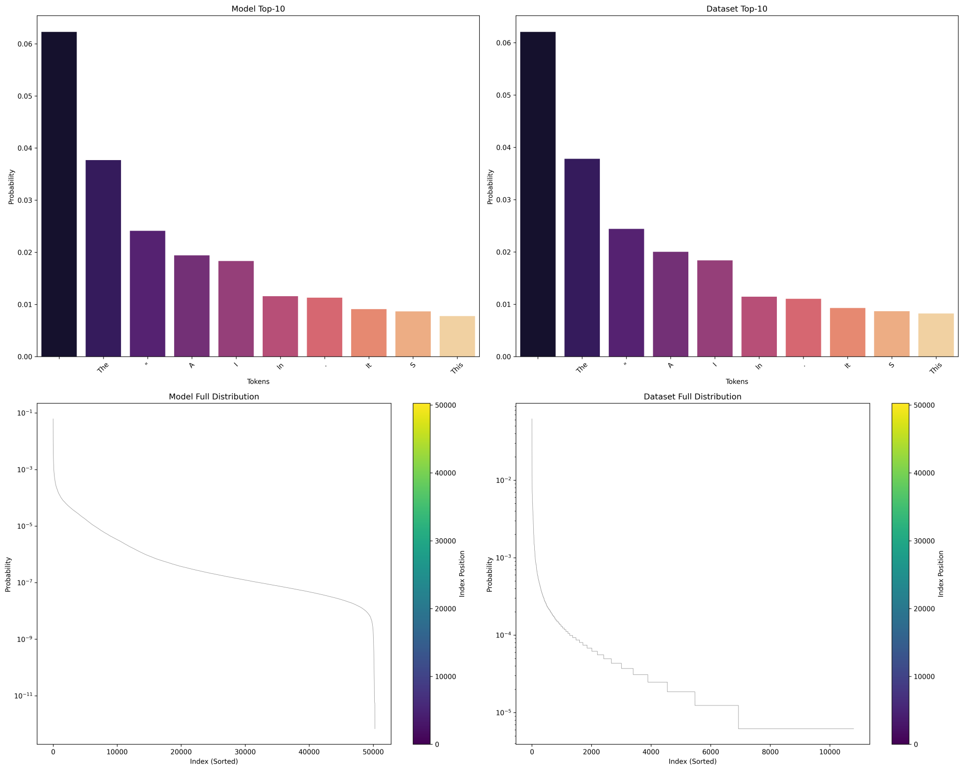

Figure 11: This figure compares the token distribution of the generated dataset for GPT-2 with the model's expected token distribution.

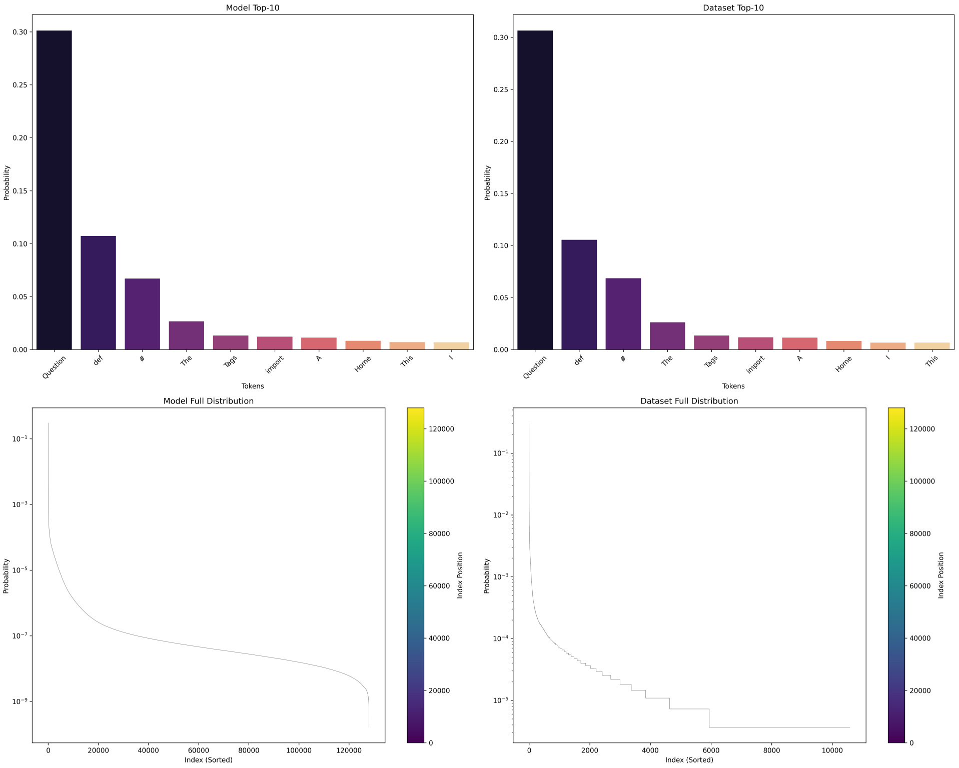

Figure 12: This figure compares the token distribution of the generated dataset for LLaMA 1B with the model's original token distribution.

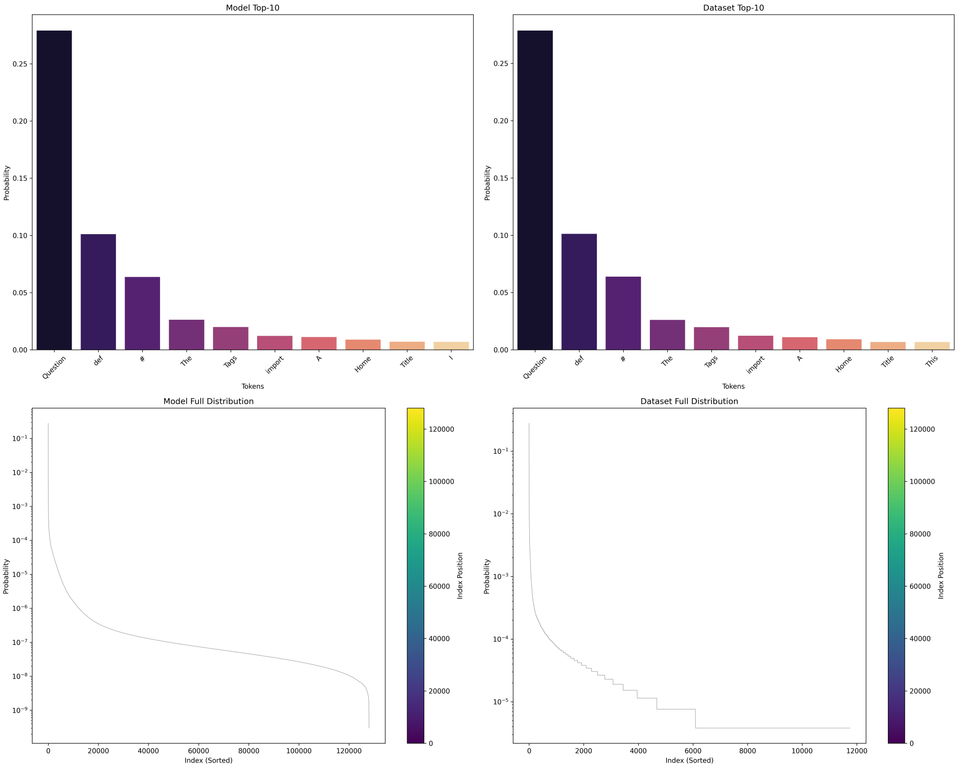

Figure 13: This comparison shows the token distribution of LLaMA 3B's generated dataset versus the model's distribution.

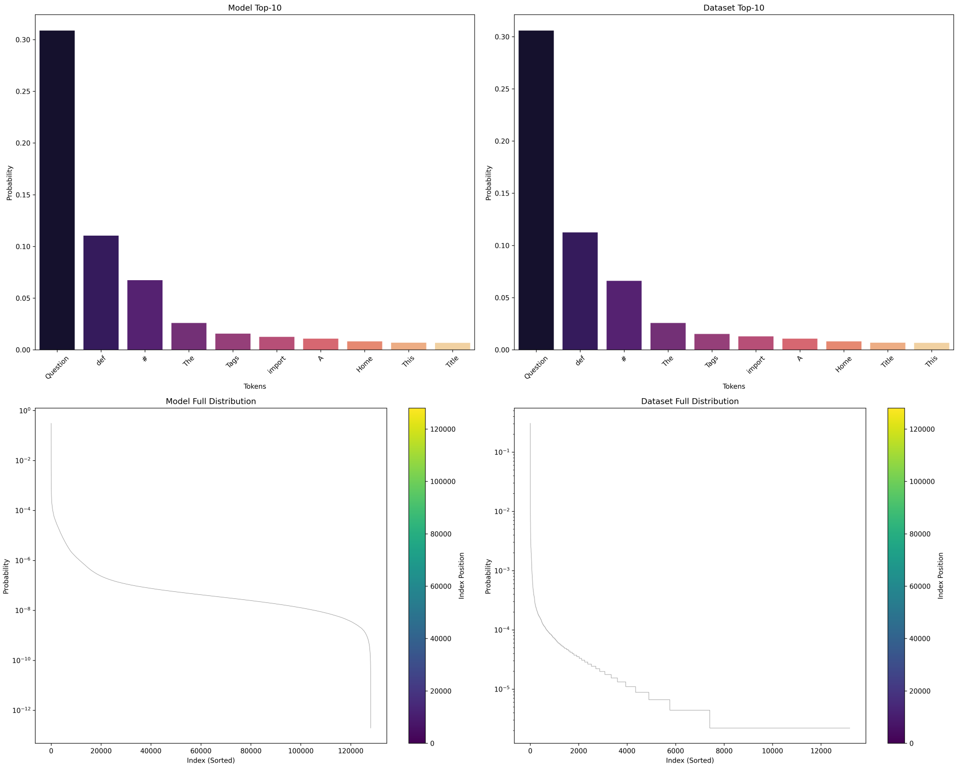

Figure 14: This figure visualizes how well the generated dataset represents LLaMA 8B's token distribution.

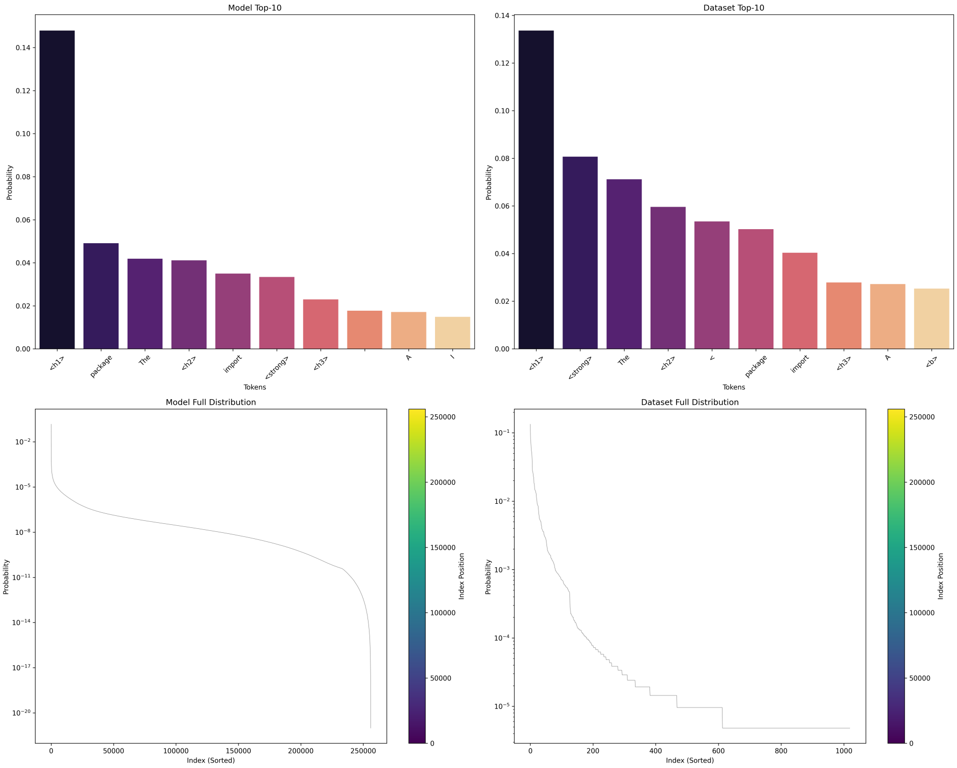

Figure 15: This visualization compares the generated token distribution with the original model for Gemma 2B.

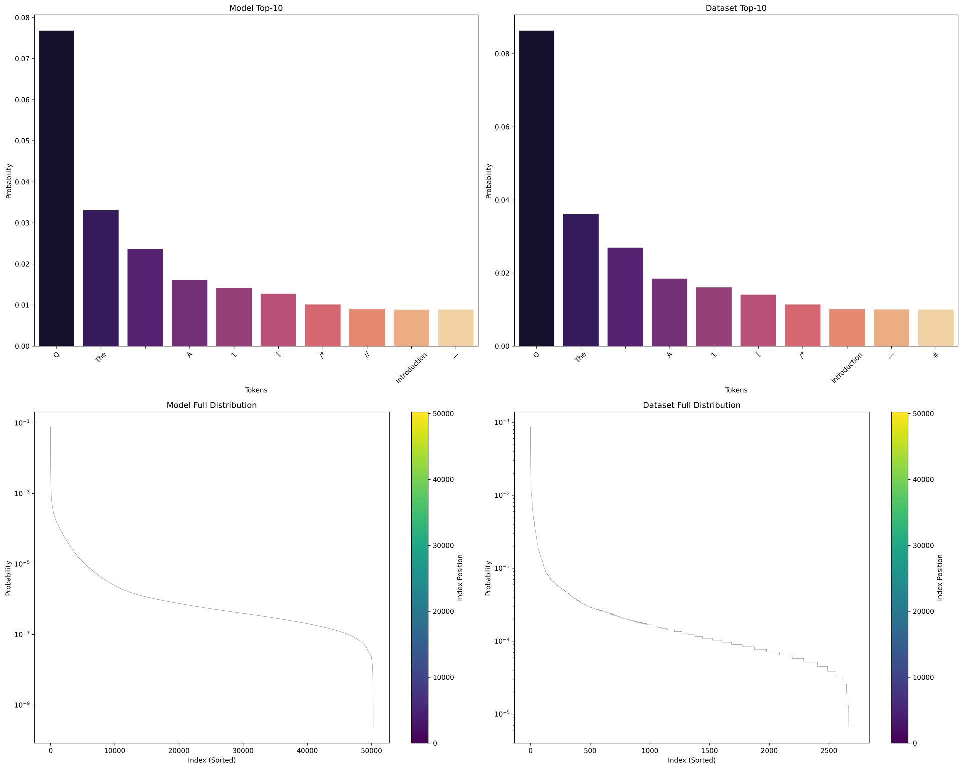

Figure 16: This figure shows the token distribution for the generated Pythia 1.4B dataset, comparing it to the model's distribution.

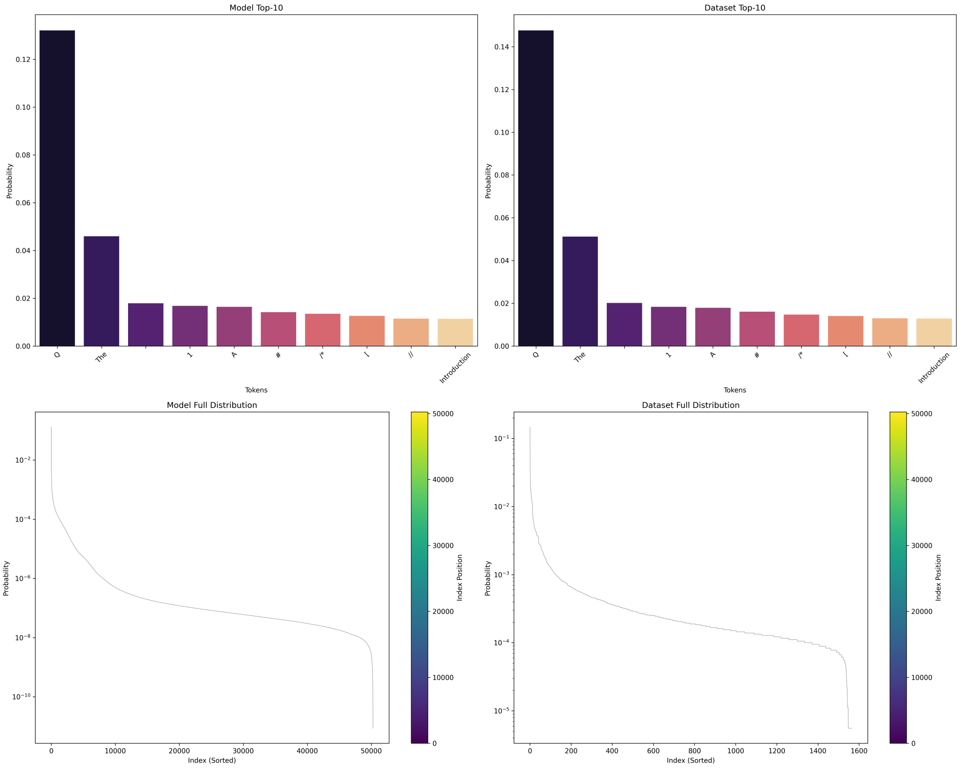

Figure 17: This figure shows the token distribution for the generated Pythia 2.8B dataset, comparing it to the model's distribution.

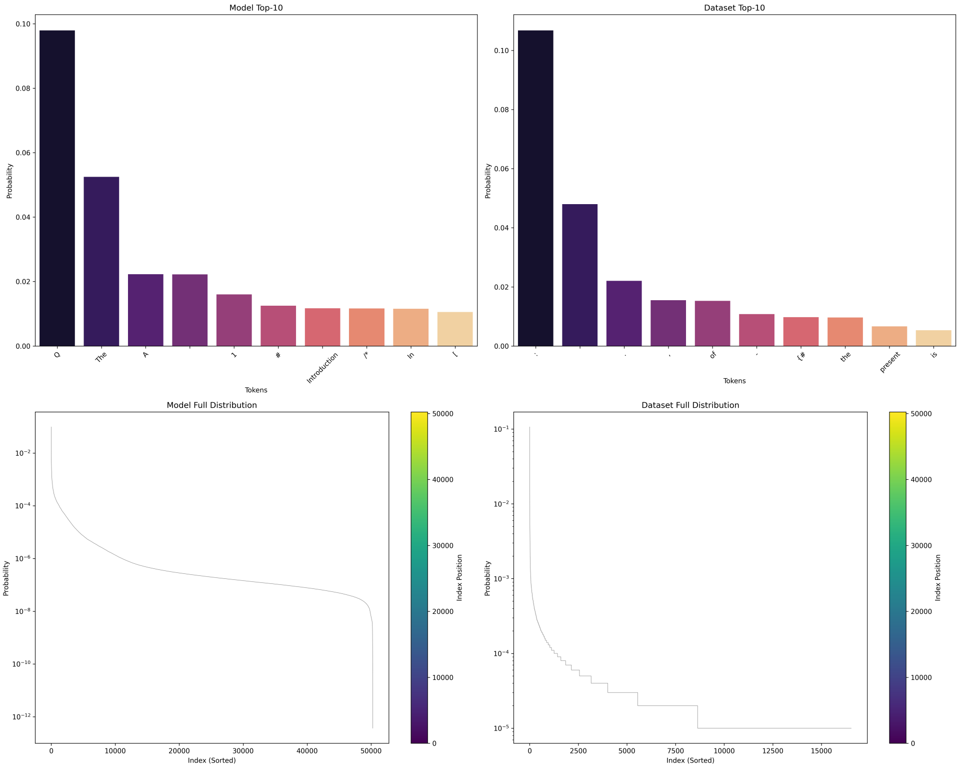

Figure 18: This figure shows the token distribution for the generated Pythia 6.9B dataset, comparing it to the model's distribution.

837 ## C.1 SAE Probing

Model	SST-2			CoLA			Yelp		
	Faithful	Fineweb	Pile	Faithful	Fineweb	Pile	Faithful	Fineweb	Pile
GPT2-small	**0.7746**	0.7723	0.7500	**0.7076**	0.6989	0.6912	**0.6532**	0.6502	0.6444
Pythia 1.4B	**0.8451**	0.8354	0.8314	**0.7281**	0.7253	0.7262	0.9341	**0.9399**	0.9289
Gemma 2B	0.7729	**0.8394**	0.8085	**0.7478**	0.7291	0.7430	**0.9536**	0.9495	0.9440
Pythia 2.8B	0.8050	0.8256	**0.8365**	**0.6985**	0.6371	0.6783	0.9392	0.9428	**0.9442**
LLaMA 1B	0.8342	**0.8491**	0.8428	**0.7469**	0.7411	0.7411	0.9431	**0.9437**	0.9429
LLaMA 3B	**0.8532**	0.8423	0.8497	**0.6889**	0.6826	0.6888	**0.9547**	0.9544	0.9525

Table 6: Reconstruction accuracy of SAE probing across 3 datasets and 6 model architectures. FaithfulSAE compared against SAEs trained on web-based datasets (Fineweb, Pile).

838 ## C.2 Fake Feature

Dataset	**GPT2**	**Pythia 1.4B**	**Gemma 2B**	**Pythia 2.8B**	**LLaMA 1B**	**LLaMA 3B**	**LLaMA 8B**
Faithful	**0.1139**	0.3871	**0.5425**	0.4655	**0.0314**	**0.1899**	**0.4150**
Pile	0.1180	0.3871	0.5669	0.4460	0.0446	0.2930	0.5341
Fineweb	0.1587	**0.3802**	0.5995	**0.4362**	0.0600	0.2713	0.5493

Table 7: Average fake feature ratio (%) across training datasets and model architectures.