Accelerating Sparse Autoencoder Training via Layer-Wise Transfer Learning in Large Language Models

Davide Ghilardi[1] *, Federico Belotti[1] *, Marco Molinari[2, 4] *, Jaehyuk Lim[2, 3]

[1]University of Milan-Bicocca, [2]LSE.AI, [3]University of Pennsylvania, [4]London School of Economics
* Equal contribution

Correspondence: d.ghilardi@campus.unimib.it

Abstract

Sparse AutoEncoders (SAEs) have gained popularity as a tool for enhancing the interpretability of Large Language Models (LLMs). However, training SAEs can be computationally intensive, especially as model complexity grows. In this study, the potential of transfer learning to accelerate SAEs training is explored by capitalizing on the shared representations found across adjacent layers of LLMs. Our experimental results demonstrate that fine-tuning SAEs using pre-trained models from nearby layers not only maintains but often improves the quality of learned representations, while significantly accelerating convergence. These findings indicate that the strategic reuse of pre-trained SAEs is a promising approach, particularly in settings where computational resources are constrained.

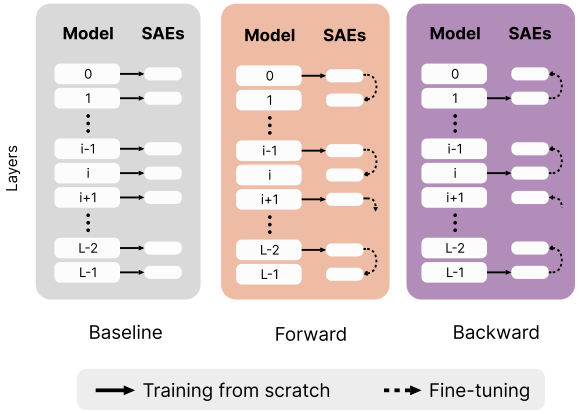

Figure 1: Visualization of our method. From left to right: **baseline** method where each Sparse AutoEncoder (SAE) is trained from scratch (solid line); **forward** method where SAEs are initialized with weights from the previous layer's SAE and fine-tuned (dashed line) with the new layer activations; **backward** method where SAEs are initialized with weights from the following layer's SAE.

1 Introduction

Transformer-based models have become ubiquitous in a large variety of different application fields (Dubey et al., 2024; Kirillov et al., 2023; Radford et al., 2023; Chen et al., 2021; Zitkovich et al., 2023; Waisberg et al., 2023). Given their tremendous impact on society, concerns about their interpretability have been raised by various stakeholders (Bernardo, 2023). Mechanistic Interpretability (MI) (Conmy et al., 2023; Nanda et al., 2023), seeks to reverse-engineer how Neural Networks, and in particular LLMs, generate outputs by uncovering the circuits they have learned during training, stored inside their parameters, and executed during a forward pass (Nanda et al., 2023; Conmy et al., 2023; Gurnee et al., 2023). A promising interpretability technique is dictionary learning (Cunningham et al., 2023; Gao et al., 2024; Karvonen et al., 2024) which seeks to capture interpretable and editable features within the internal layers of LLMs. This method implies training Sparse Autoencoders (SAEs) to reconstruct the model's activations using sparse learned features. However, training SAEs is computationally intensive, particularly when applied across multiple layers in deep networks. This computational burden poses a significant barrier to their widespread application, especially in resource-constrained environments where the cost of training from scratch is prohibitive. Recent research has highlighted the potential of transfer learning as a strategy to mitigate these challenges (Kissane et al., 2024). In particular, it has been shown in Gromov et al. (2024) that adjacent layers in LLMs are often redundant, suggesting that the knowledge encoded in one layer is also present in neighboring ones and that it can effectively be transferred. This observation forms the basis of our investigation: we hypothesize that SAEs trained on one set of layers can serve as effective initialization for SAEs designed for closely related layers. Specifically, the *forward* approach is defined as initializing an SAE with the weights of

a previous layer SAE, and the *backward* approach as initializing an SAE with the weights of a subsequent layer SAE. The overall training procedure is summarized in Figure 1. We tested this hypothesis on Pythia-160M, a small 12-layer decoder-only transformer from the Pythia family (Biderman et al., 2023). By reusing the representations learned in earlier layers, computational demands of training can be reduced by at least 25%[1] while maintaining, or even improving, the quality of the resulting models. Our contributions are as follows:

- We demonstrate that SAEs exhibit partial transfer to adjacent layers in a zero-shot setting, though fine-tuning is recommended for optimal performance.
- We show that both Forward-SAEs and Backward-SAEs, when fine-tuned on adjacent activations, consistently transfer across all tested checkpoints, achieving comparable or superior performance to SAEs trained from scratch, while using significantly less training data.
- We train and publicly release SAEs for Pythia-160M (Biderman et al., 2023), the model utilized in this study.

Code, data, and trained models will be publicly released after the double-blind review.

2 Background and objectives

2.1 Linear representation hypothesis and superposition

Although it has been demonstrated that LLMs represent some of their feature linearly (Park et al., 2024), a key challenge in LLM interpretability is the lack of clear neuron interpretation. Recent work of Elhage et al. (2022) tries to explain this phenomenon by showing that models can use n-dimensional activations to represent $m \gg n$ sparse almost-orthogonal features in *superposition*. Superposition theory is based on three key concepts: (i) the existence of a hypothetical large and disentangled model where each neuron perfectly aligns with a single feature, with each neuron activating for exactly one feature at a time. The observed models can be thought as dense, almost-orthogonal projections of this larger, ideal model. (ii) Features are sparse, reflecting the idea that in the natural world, many features are inherently sparse. (iii) The importance of features varies depending on the task at hand. These assumptions, combined with two mathematical principles[2], suggest that the hidden sparse features can be recovered by projecting the dense model back to the hypothetical large and disentangled one. SAEs serve this purpose: learning a set of sparse, interpretable, and high-dimensional features from an observed model's dense and superposed activations.

2.2 Sparse Autoencoders

Recently, Sparse AutoEncoders have become a popular tool in Large Language Model (LLM) interpretability as they effectively decompose neuron activations into interpretable features (Bricken et al., 2023; Cunningham et al., 2023). For a given input activation $\mathbf{x} \in \mathbb{R}^{d_{\text{model}}}$, the SAE computes a reconstruction $\hat{\mathbf{x}}$ as a sparse linear combination of $d_{\text{sae}} \gg d_{\text{model}}$ features $\mathbf{v}_i \in \mathbb{R}^{d_{\text{model}}}$. The reconstruction is given by:

$$(\hat{\mathbf{x}} \circ \mathbf{f})(\mathbf{x}) = \mathbf{W}_d \mathbf{f}(\mathbf{x}) + \mathbf{b}_d \tag{1}$$

where $\mathbf{v}_i$ are the columns of $\mathbf{W}_d$, $\mathbf{b}_d$ is the bias term of the decoder and $\mathbf{f}(\mathbf{x})$ are feature activations. The latter are computed as:

$$\mathbf{f}(\mathbf{x}) = \text{ReLU}(\mathbf{W}_e(\mathbf{x} - \mathbf{b}_d) + \mathbf{b}_e) \tag{2}$$

where $\mathbf{b}_e$ is the encoder bias term. SAEs are trained to minimize the following loss function:

$$\mathcal{L}_{\text{sae}} = \|\mathbf{x} - \hat{\mathbf{x}}\|_2^2 + \lambda \|\mathbf{f}(\mathbf{x})\|_1 \tag{3}$$

In Equation 3, the first term corresponds to the reconstruction error, to which an ℓ_1 regularization term on the activations $\mathbf{f}(\mathbf{x})$ is added to promote sparsity in the feature activations. In SAEs training, it is common to set $d_{\text{sae}} = c\, d_{\text{model}}$ with $c \in \{2^n \mid n \in \mathbb{N}_+\}$. So, the training process of a SAE can become computationally intensive, particularly as model size increases. For example, training a single SAE of a widely used model such as Llama-3-8b (Dubey et al., 2024) ($d_{\text{model}} = 4096$) with an expansion factor of $c = 32$ (i.e., $d_{\text{sae}} = 131072$) requires $\approx$ 1B parameters. Under these circum-

[1] Assuming training half of SAEs from scratch and the other half with transfer from an adjacent layer with half of the training tokens.

[2] The Johnson-Lindenstrauss lemma, which ensures that points in a high-dimensional space can be embedded into a lower dimension while almost preserving distances, and compressed sensing, which exploits sparsity to recover signals from fewer samples than required by the Nyquist–Shannon theorem

Config	Value
Layers (L)	12
Model dimension (d_{model})	768
Heads (H)	12
Non-Embedding params	85,056,000
Equivalent models[3]	GPT-Neo OPT-125M

Table 1: Pythia-160M model specifics

stances, transfer learning is a useful resource to reduce the number of trained SAEs, with the transfer that can happen *intra-model*, where SAEs training is shared between layers of the same model (our case), or *inter-model*, where SAEs are shared between different fine-tuned versions of the same model as shown in Kissane et al. (2024).

2.3 Evaluating SAEs

Evaluating SAEs and the features they have learned presents significant challenges. In our work, the techniques employed can be divided into *reconstruction* and *interpretability* metrics. The first includes:

- The Cross-Entropy Loss Score (CES), is defined as

$$\text{CES} = \frac{\text{CE}(\zeta) - \text{CE}(\hat{\mathbf{x}} \circ \mathbf{f})}{\text{CE}(\zeta) - \text{CE}(\text{Id})} \tag{4}$$

  where $\hat{\mathbf{x}} \circ \mathbf{f}$ is the autoencoder function, $\zeta : \mathbf{x} \to \mathbf{0}$ the zero-ablation function and $\text{Id} : \mathbf{x} \to \mathbf{x}$ the identity function. According to this definition, a SAE would get a CES equal to 1 if it perfectly reconstructs $\mathbf{x}$ (> 1 if it improves the CE loss), ≤ 0 when the reconstruction is not better than zero-ablation, otherwise the score is comprised in the unit interval.

- The $L2$ loss (reconstruction loss) is the first term of Equation 3, which measures the reconstruction error made by the SAE.

- The $L0$ loss of the learned features, defined as

$$\|\mathbf{f}\|_0 = \sum_{j=1}^{|\mathbf{f}|} \mathbb{I}[\mathrm{f}_j \neq 0] \tag{5}$$

  which represents the number of non-zero SAE features used to compute the reconstruction.

Measuring the quality of the features learned by a SAE is not straightforward, and multiple strategies exist. As reported in Makelov et al. (2024), *interpretability* metrics can be categorized as follows:

- Indirect Geometric Measures: Sharkey et al. (2023) proposed using mean maximum cosine similarity (MMCS) between features learned by different SAEs to assess their quality. Given two feature dictionaries D and D', with $|D| = |D'|$, MMCS is defined as:

$$\text{MMCS}_{D,D'} = \frac{1}{|D|} \sum_{\mathbf{u} \in D} \max_{\mathbf{v} \in D'} \text{CosSim}(\mathbf{u}, \mathbf{v}) \tag{6}$$

- Auto-Interpretability: Bricken et al. (2023), Bills et al. (2023), and Cunningham et al. (2023) used LLMs to generate natural-language descriptions of SAE features based on highly activating examples and measured interpretability as the prediction quality on previously unseen text.

- Manually Crafted Proxies for Ground Truth: (Bricken et al., 2023) developed computational proxies for a set of SAE features, relying on manually formulated hypotheses.

- Faithfulness and Completeness of task feature circuits: Marks et al. (2024) compute faithfulness and completeness as measures to estimate the task sufficiency and necessity of learned SAE features. In particular, given a task, they first compute a circuit C of SAE features by selecting them according to their importance, estimated via their Indirect Effect[4] (Pearl, 2022):

$$\begin{aligned}&\text{IE}(m; \mathbf{f}; a_c, a_w) = \\ &\quad m[M(a_c|\text{do}(\mathbf{f} = \mathbf{f}_w), x); M(a_c|x)]\end{aligned} \tag{7}$$

  where x is a given prompt and $m : \mathbb{R}^{d_{\text{vocab}}} \to \mathbb{R}$ is the logit-difference computed by a LLM M over two contrastive answer tokens a_c, a_w.[5] In this equation, $\mathbf{f}_w$ represents SAE feature activations during the computation of

[3] As specified in (Biderman et al., 2023)

[4] We estimate the IE through Attribution Patching (AtP) (Syed et al., 2023; Nanda, 2023) A formal definition of AtP is given in Appendix A

[5] E.g., x = "The square root of 9 is", $a_c = 3$, and $a_w = 2$

$M(a_w|x)$, and $M(a_c|\text{do}(\mathbf{f} = \mathbf{f}_w), x)$ refers to the value of $M(a_c)$ under an intervention where the activation of feature $\mathbf{f}$ is set to $\mathbf{f}_w$. Then, they estimate the *faithfulness* as

$$\frac{m(C) - m(\emptyset)}{m(M) - m(\emptyset)} \tag{8}$$

where $m(C)$ is the model logit difference when using only the important SAE features while mean-ablating the others; $m(M)$, $m(\emptyset)$ represent the logit-difference achieved by the model alone and with the mean-ablated SAE reconstructions, respectively. *Completeness* is estimated by replacing $m(C)$ with $m(M \setminus C)$ in Equation 8. Intuitively, faithfulness captures the proportion of the model's performance the circuit C explains, relative to mean-ablate the full model, thus modeling sufficiency. On the other hand, completeness captures the necessity of the learned features by measuring low downstream performance whenever the important SAE features are mean-ablated.

- Supervised Dictionary Benchmarking: Makelov et al. (2024) introduced a technique that benchmarks unsupervised SAE dictionaries against supervised dictionaries based on task-relevant attributes to ensure extracted features are interpretable and relevant to specific tasks.

In our work, evaluation metrics employed include all the reconstruction techniques listed above, the MMCS between features from SAEs trained with transfer learning and the ones from SAEs trained from scratch, and a Human Interpretability Score defined in Section 3. Moreover, we evaluate both faithfulness and completeness on three standard downstream tasks: Indirect Object Identification (IOI) (Wang et al., 2023), Greater Than (Hanna et al., 2023), and Subject-Verb Agreement (Marks et al., 2024), all of them comprising a set of examples in the form of $\{(x, a_c, a_w)_i\}$. Additionally, for faithfulness and completeness computation we fix the number of top important features N throughout all the experiments: for faithfulness we let N vary in $\{123, 246, 368, 492\}$, which correspond to $2\%, 4\%, 6\%$ and 8% of top active features; for completeness, N varies in $\{4, 36, 68, 100\}$.[6] Finally, in Appendix B we report the Direct Logit Attribution (DLA), as specified by Bricken et al. (2023).

[6]Top important features are computed on a per-example basis.

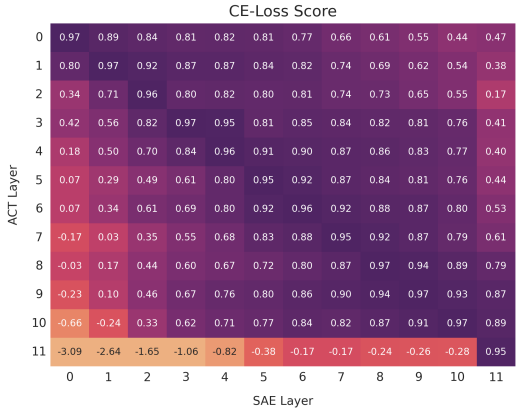

Figure 2: Cross-Entropy Loss Score (CE-Loss Score) (Eq. 4), where the cell (i, j) in the plot represents the CE-Loss Score obtained by reconstructing the activations from layer i with SAE_j. This plot has to be read column-wise.

2.4 Transfer Learning

Transfer learning (Goodfellow et al., 2016) is a powerful technique in machine learning where knowledge gained from one task is applied to improve performance on a related, but distinct, task. This approach is particularly useful when training from scratch is computationally expensive or when labeled data is scarce. In the context of SAEs for LLMs, transfer learning enables the reuse of weights learned in one layer to initialize and accelerate the training of SAEs in adjacent layers.

2.5 Objectives

In this work transferability and generalization of intra-model SAEs have been studied, aiming to answer the following research questions:

Q1. Are SAEs transferable between layers? I.e., can a SAE trained on the activations of layer i be reused to reconstruct activations of layer $j \neq i$?

Q2. Is Transfer Learning applicable to SAEs? Specifically, can a SAE initialized with the weights of a neighboring SAE and then fine-tuned achieve equal or superior performance, potentially using only a fraction of the data, compared to an SAE trained from scratch?

Figure 3: Average CE-Loss Score, $L2$-Loss and $L0$-Loss. The average is computed over layers for a single checkpoint. The "No Transfer" average is computed considering the performance obtained by $\text{SAE}_i(\mathbf{x}_i), \forall i = 0, ..., 11$.

3 Experimental setup

To address the questions raised in Section 2, we first trained from scratch one SAE_i for each layer i of Pythia-160M, a 12-layer decoder-only Transformer model from the Pythia family (Biderman et al., 2023). Each SAE was trained using the JumpReLU activation function (Rajamanoharan et al., 2024), with activations taken from the corresponding layer's residual stream after the MLP contribution. The model configuration details are provided in Table 1. Let also $j \neq i$ be another layer index. Then $\text{SAE}_{i \leftarrow j}$ is defined as the SAE initialized with weights from the j-th SAE and fine-tuned with activations of the i-th layer. In particular, this work is focused on $\text{SAE}_{i \leftarrow i-1}$ and $\text{SAE}_{i \leftarrow i+1}$, named Forward-SAE (Fwd-SAE) and Backward-SAE (Bwd-SAE) respectively. Figure 1 summarizes the overall training and fine-tuning procedure, with the hyperparameters specified in Table 2. The dataset adopted for both training and fine-tuning is the Pile-small-2b[7], an already tokenized version of the Pile dataset (Gao et al., 2020) with a total of 2b tokens. To effectively measure the reconstruction performance of a SAE before and after fine-tuning with transfer learning, the normalized CE-Loss Score is adopted and defined as:

$$\overline{\text{CES}_{i,j}} = \frac{\text{CES}(\text{SAE}_{i \leftarrow j}(\mathbf{x}_i)) - \text{CES}(\text{SAE}_j(\mathbf{x}_i))}{\text{CES}(\text{SAE}_i(\mathbf{x}_i)) - \text{CES}(\text{SAE}_j(\mathbf{x}_i))} \tag{9}$$

by assuming $\text{CES}(\text{SAE}_j(\mathbf{x}_i))$ and $\text{CES}(\text{SAE}_i(\mathbf{x}_i))$ being, respectively, the lower and the upper bound for the CES on $\mathbf{x}_i$. With the definitions above, $\overline{\text{CES}_{i,i-1}}$ and $\overline{\text{CES}_{i,i+1}}$ are the normalized CE-Loss Score of the Fwd-SAE and Bwd-SAE respectively. Finally, to evaluate feature quality, a *Human Interpretability Score* has been defined as the ratio of features that have been evaluated interpretable by human annotators. To generate the score, all the SAEs have been run on approximately 1M tokens randomly sampled from the training dataset. With their activations, max activating tokens and top/bottom attribution logits have been computed and analyzed from the labelers.

[7] https://huggingface.co/datasets/NeelNanda/pile-small-tokenized-2b

4 Results

4.1 SAE transferability

Figure 2 shows the CE-Loss Score achieved by every SAE_j reconstructing the activations of layer i, for every $i, j = 0, ..., L-1$, i.e., the zero-shot setting. It is clear that a certain degree of transferability exists between SAE_j and the activations of adjacent layers, with this being more noticeable when $i = j - 1$ (i.e., SAEs are more effective at reconstructing the activations of preceding layers than those of subsequent ones). These findings can also be attributed to the fact that, as demonstrated by Gromov et al. (2024), angular distances between adjacent layers are smaller, enabling neighboring SAEs to operate on a similar basis with respect to the activations they were trained on. The answer to **Q1** is, therefore, yes; however, although transferability between layers exists, it remains partial and, potentially, not completely reliable for downstream applications.

4.2 SAE transfer learning

Figure 3 shows all reconstruction metrics averaged for all layers across every tested checkpoint. Detailed results for single layer and aggregated over time can be found in Appendix C (Figures 9 - 17) along with the normalized CE-Loss Score

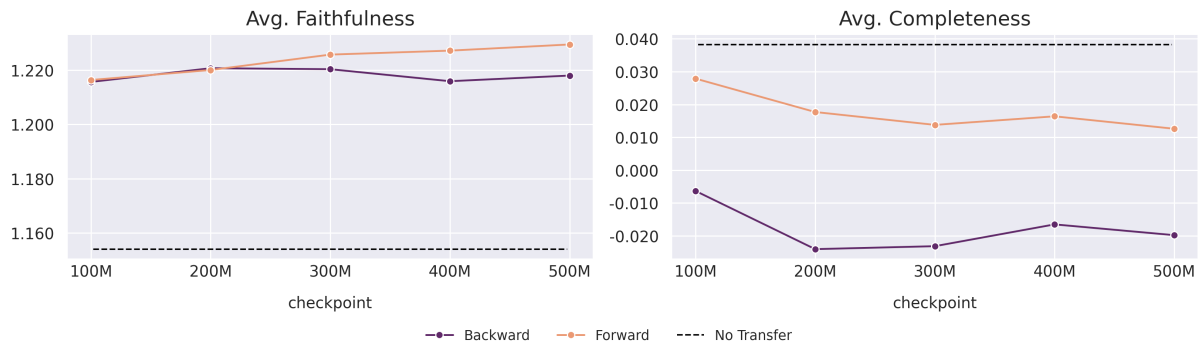

Figure 4: Average Faithfulness and Completeness. The average is computed over layers and the number of important active SAE features for a single checkpoint. The "No Transfer" average is computed considering the performance obtained by $\text{SAE}_i(\mathbf{x}_i), \forall i = 0, ..., 11$.

(Eq. 9) in Tables 3 and 4. Looking at the plots, it can be seen that forward and backward SAEs achieve almost equal or even superior performance than the ones trained from scratch with as little as $1/10$-th (100M tokens) of the original training data (1B tokens), with the scores constantly increasing with the number of tokens used for fine-tuning. As a result, it can be said that both forward ad backward are effective strategies to reduce the number of SAEs trained from scratch. Between the two, the backward technique is the one that constantly shows better results, both in terms of CE-Loss Score, $L2$, and $L1$ loss. So, the answer to **Q2** is also yes if we just consider the reconstruction metrics. To fully respond to **Q2** beyond reconstruction performance, the quality of the learned SAE features have to be inspected.

4.3 Feature Evaluation

Figure 4 displays the layer-averaged faithfulness and completeness scores for each tested checkpoint. The plot reveals that both forward and backward

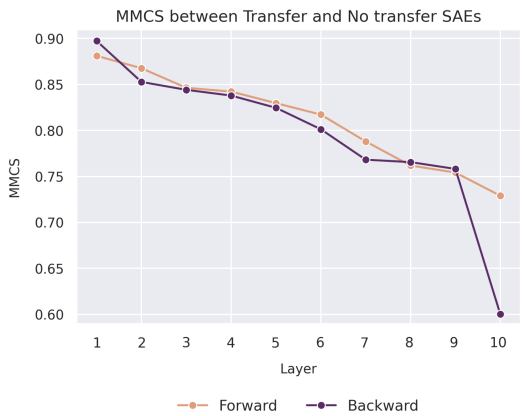

Figure 5: Per-layer MMCS of the Forward and Backward SAEs.

transfer SAEs consistently achieve better scores than the baseline SAEs, with minimal differences between the two transfer methods. Therefore, both the forward and backward SAEs maintain sufficiency and necessity during their transfer. Figure 5 presents the MMCS between SAEs trained with transfer learning and those trained from scratch. The metric value decreases for deeper layers, suggesting a slight divergence in the features learned by the transfer SAEs. Notably, $\text{SAE}_{L-1 \leftarrow L}$ exhibits a sharp decline in the score, indicating that transferring on the last layer should be approached with caution. Lastly, from human interpretability scores (Figure 7), no significant differences can be observed between each transfer type. By manually looking at the learned features, a key pattern has emerged: many features learned by SAEs trained with transfer learning remain shared with the SAE used for initialization. This phenomenon, termed *Feature Transfer*, particularly affects the most interpretable features (see an example in Figure 23). To further investigate this phenomenon, a metric was developed to quantify it. Given a SAE_i and another trained via transfer learning from it, $\text{SAE}_{i \leftarrow i \pm 1}$, the number of shared "top", "bottom", and "max activating tokens"[8] for each feature have been computed (features have been compared using the same indices). The transfer score has been then defined as the percentage of shared tokens across all three heuristics. Figure 6 presents the scores across all the layers for the last evaluated checkpoint. Except for layer 1, backward transfer consistently exhibits lower scores. It's important to note that this phe-

[8] "Top" and "bottom" logit tokens refer to those whose unembedding directions are most and least aligned, respectively, with the projection of the feature in the unembedding space. "Max activating" tokens are those for which the feature exhibits the highest activations.

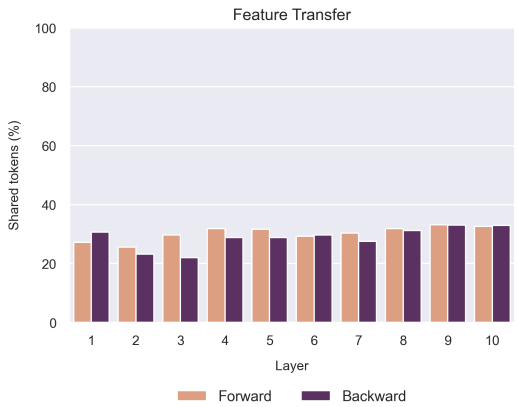

Figure 6: Per-layer number of shared tokens for the Forward and Backward SAEs, as defined in Section 4.3. Each bar represents the percentage of shared token between SAE_i trained from scratch and forward $\text{SAE}_{i+1\leftarrow i}$ and backward $\text{SAE}_{i-1\leftarrow i}$, respectively.

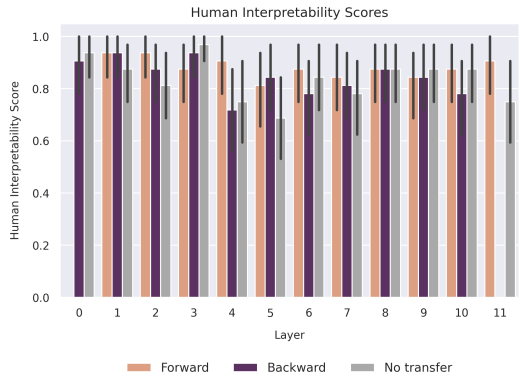

Figure 7: Human Interpretability Scores (Section 3) for 32 features randomly sampled from each SAE layer and type of transfer.

nomenon is easily recognized in SAEs trained with transfer learning when compared to their initialization, as feature indices are preserved. Evaluating this in SAEs trained from scratch is more demanding due to the exponential growth in the number of comparisons required, and although relevant, it falls outside the scope of this work.

4.4 Compute Efficiency

Leveraging forward and backward transfer, we were able to reduce total training steps when utilizing forward transfer and backward transfer by 42% and 46%, respectively. Check Appendix B.1 for details.

5 Related works

5.1 Scaling and evaluating SAEs

As SAEs gain popularity for LLMs interpretability and are increasingly applied to state-of-the-art models (Lieberum et al., 2024), the need for more efficient training techniques has become evident. To address this, Gao et al. (2024) explored scaling laws of autoencoders to identify the optimal combination of size and sparsity. However, training SAEs is only one aspect of the challenge; evaluating them presents another significant hurdle. This evaluation is a crucial focus within MI. While early approaches in Cunningham et al. (2023) and (Bricken et al., 2023) relied on unsupervised metrics like reconstruction loss and $L0$ sparsity to assess SAE performance, these metrics alone cannot fully capture the efficacy of a SAE. They provide quantitative measures of how well SAEs capture information in model activations while maintaining sparsity, but they fall short of addressing the broader utility of these features. More recent techniques, such as auto-interpretability (Bricken et al. (2023), Bills et al. (2023), Cunningham et al. (2023)) and ground-truth comparisons (Sharkey et al., 2023), have shifted towards a more holistic evaluation, focusing on the causal relevance of the extracted features (Marks et al., 2024) and evaluating SAEs on different downstream tasks in which they can be employed (Makelov et al., 2024). In particular, Makelov et al. (2024) introduced a framework for evaluating SAEs on the Indirect Object Identification (IOI) task, focusing on three key aspects: the sufficiency and necessity of activation reconstructions, the ability to control model behavior through sparse feature editing, also called feature steering (Templeton et al., 2024), and the interpretability of features in relation to their causal role. Karvonen et al. (2024) further advanced principled evaluations by introducing novel metrics specifically designed for board game language models. Their approach leverages the well-defined structure of chess and Othello to create supervised metrics for SAE quality, including board reconstruction accuracy and coverage of predefined board state properties. These methods provide a more direct assessment of how well SAEs capture semantically meaningful and causally relevant features, offering a complement to the earlier unsupervised metrics like $L0$ and $L2$.

5.2 SAEs transfer learning

Recent work by Kissane et al. (2024) and Lieberum et al. (2024) has demonstrated the transferability of SAE weights between base and instruction-tuned

versions of the Gemma-1 (Team et al., 2024a) and Gemma-2 (Team et al., 2024b), respectively. This finding is significant as it suggests that many interpretable features are preserved during the fine-tuning process. While this transfer occurs between model variants (inter-model) rather than between layers (intra-model), it complements our work by indicating that SAE features can remain stable across different stages of model development. The preservation of these features through fine-tuning not only offers insights into the robustness of learned representations but also suggests potential efficiency gains in interpreting families of models derived from a common base SAE.

6 Conclusions

We hypothesized and validated whether SAE transfer is an effective method to accelerate and optimize the SAE training process. We investigated whether SAE weights derived from adjacent layers could maintain efficacy in reconstruction, which our results affirmed. Furthermore, we examined whether the transferred SAEs, when fine-tuned on a layer's activations, could reliably capture monosemantic features comparable to the original SAE, which has been also confirmed by our experiments. The transferred SAEs (both forward and backward) demonstrated comparable and occasionally superior reconstruction loss relative to the original. Empirically, we observed frequent overlap in the most strongly activated features across adjacent layers (e.g. Figure 23). For a given feature index i, the features learned by $\text{SAE}_{i \leftarrow i+1}$ (Backward), SAE_i (No Transfer), and $\text{SAE}_{i \leftarrow i-1}$ (Forward) appeared to represent similar concepts.

7 Limitations and future works

While our study successfully demonstrates the feasibility of reconstruction transfer and the transfer learning of SAE weights to adjacent layers, there are several limitations that warrant consideration and pave the way for future research directions.

- *Model Size and Scope*: We trained base and transfer SAEs on the activations of Pythia-160m, a model mcuh smaller than state-of-the-art LLMs. Although not being tested, as model size and training complexity increase, the benefits of transfer learning are expected to become more pronounced. In such scenarios, transfer learning can significantly accelerate training and reduce associated costs, making our approach potentially more impactful for larger models. Therefore, a critical area for future research is to extend these investigations to larger models, exploring how scaling affects the efficacy of transfer learning and how these benefits can be maximized in real-world settings.

- *Inter-Model and Intra-Model transferability*: In our study, we focused on the transfer of intra-model SAEs, particularly assessing the transferability between SAEs in adjacent layers. Given that model architectures are now commonly shared across different model families, a direction for future research would be to evaluate the transferability of intra-model SAEs within models from different families that utilize the same architecture. This exploration could offer valuable insights into the broader applicability of SAEs beyond closely related model families.

- *Experimental Scale and Hyperparameter Interactions*: Our study was conducted on a limited scale in terms of model components involved and the range of training hyperparameters explored. The fixed set of hyperparameters used may not fully capture the potential of our transfer learning approach across different configurations. Future research should involve a broader exploration of hyperparameter spaces, especially the λ coefficient and expansion factor c, along with component variations to determine the robustness and versatility of the method.

- *Feature Transfer Phenomenon*: Our findings reveal a "feature transfer" phenomenon, where features learned in one layer are exactly replicated in another during transfer learning. This can be problematic, as it may prevent the fine-tuned SAEs from discovering new, layer-specific features. However, it also offers an interesting opportunity to study how similar features are encoded across layers. Future research should focus on understanding and managing this phenomenon to either harness or mitigate its effects, depending on the desired outcomes, thereby improving the flexibility and effectiveness of transfer learning.

Acknoledgements

This work has been partially funded by the European innovation action enRichMyData (HE 101070284).

References

Vítor Bernardo. 2023. Techdispatch #2/2023 - explainable artificial intelligence. https://www.edps.europa.eu/data-protection/our-work/publications/techdispatch/2023-11-16-techdispatch-22023-explainable-artificial-intelligence_en. European Data Protection Supervisor.

Stella Biderman, Hailey Schoelkopf, Quentin Gregory Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, et al. 2023. Pythia: A suite for analyzing large language models across training and scaling. In *International Conference on Machine Learning*, pages 2397–2430. PMLR.

Steven Bills, Nick Cammarata, Dan Mossing, Henk Tillman, Leo Gao, Gabriel Goh, Ilya Sutskever, Jan Leike, Jeff Wu, and William Saunders. 2023. Language models can explain neurons in language models. Accessed: 2024-08-18.

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. 2023. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*. Https://transformer-circuits.pub/2023/monosemantic-features/index.html.

Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. 2021. Decision transformer: Reinforcement learning via sequence modeling. In *Advances in Neural Information Processing Systems*, volume 34, pages 15084–15097. Curran Associates, Inc.

Arthur Conmy, Augustine Mavor-Parker, Aengus Lynch, Stefan Heimersheim, and Adrià Garriga-Alonso. 2023. Towards automated circuit discovery for mechanistic interpretability. *Advances in Neural Information Processing Systems*, 36:16318–16352.

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

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whit-

ney Meers, Xavier Martinet, Xiaodong Wang, Xiaoqing Ellen Tan, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aaron Grattafiori, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alex Vaughan, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Franco, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, Danny Wyatt, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Firat Ozgenel, Francesco Caggioni, Francisco Guzmán, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Govind Thattai, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Igor Molybog, Igor Tufanov, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Karthik Prasad, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kun Huang, Kunal Chawla, Kushal Lakhotia, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Maria Tsimpoukelli, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikolay Pavlovich Laptev, Ning Dong, Ning Zhang, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Rohan Maheswari, Russ Howes, Ruty Rinott, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Kohler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaofang Wang, Xiaojian Wu, Xiaolan Wang, Xide Xia, Xilun Wu, Xinbo Gao, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yuchen Hao, Yundi Qian, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, and Zhiwei Zhao. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

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. *Preprint*, arXiv:2209.10652.

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. 2020. The pile: An 800gb dataset of diverse text for language modeling. *Preprint*, arXiv:2101.00027.

Leo Gao, Tom Dupré la Tour, Henk Tillman, Gabriel Goh, Rajan Troll, Alec Radford, Ilya Sutskever, Jan Leike, and Jeffrey Wu. 2024. Scaling and evaluating sparse autoencoders. *Preprint*, arXiv:2406.04093.

Ian J. Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press, Cambridge, MA, USA. http://www.deeplearningbook.org.

Andrey Gromov, Kushal Tirumala, Hassan Shapourian, Paolo Glorioso, and Daniel A. Roberts. 2024. The

unreasonable ineffectiveness of the deeper layers. *Preprint*, arXiv:2403.17887.

Wes Gurnee, Neel Nanda, Matthew Pauly, Katherine Harvey, Dmitrii Troitskii, and Dimitris Bertsimas. 2023. Finding neurons in a haystack: Case studies with sparse probing. *arXiv preprint arXiv:2305.01610*.

Michael Hanna, Ollie Liu, and Alexandre Variengien. 2023. How does GPT-2 compute greater-than?: Interpreting mathematical abilities in a pre-trained language model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Adam Karvonen, Benjamin Wright, Can Rager, Rico Angell, Jannik Brinkmann, Logan Riggs Smith, Claudio Mayrink Verdun, David Bau, and Samuel Marks. 2024. Measuring progress in dictionary learning for language model interpretability with board game models. In *ICML 2024 Workshop on Mechanistic Interpretability*.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. 2023. Segment anything. *Preprint*, arXiv:2304.02643.

Connor Kissane, Ryan Krzyzanowski, Andrew Conmy, and Neel Nanda. 2024. SAEs (usually) transfer between base and chat models. https://www.alignmentforum.org/posts/fmwk6qxrpW8d4jvbd/saes-usually-transfer-between-base-and-chat-models. AI Alignment Forum.

Tom Lieberum, Senthooran Rajamanoharan, Arthur Conmy, Lewis Smith, Nicolas Sonnerat, Vikrant Varma, János Kramár, Anca Dragan, Rohin Shah, and Neel Nanda. 2024. Gemma scope: Open sparse autoencoders everywhere all at once on gemma 2. *arXiv preprint arXiv:2408.05147*.

Aleksandar Makelov, George Lange, and Neel Nanda. 2024. Towards principled evaluations of sparse autoencoders for interpretability and control. *Preprint*, arXiv:2405.08366.

Samuel Marks, Can Rager, Eric J Michaud, Yonatan Belinkov, David Bau, and Aaron Mueller. 2024. Sparse feature circuits: Discovering and editing interpretable causal graphs in language models. *arXiv preprint arXiv:2403.19647*.

Neel Nanda. 2023. Attribution patching: Activation patching at industrial scale. https://www.neelnanda.io/mechanistic-interpretability/attribution-patching. Mechanistic Interpretability.

Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. 2023. Progress measures for grokking via mechanistic interpretability. In *The Eleventh International Conference on Learning Representations*.

Kiho Park, Yo Joong Choe, and Victor Veitch. 2024. The linear representation hypothesis and the geometry of large language models. *Preprint*, arXiv:2311.03658.

Judea Pearl. 2022. Direct and indirect effects. In *Probabilistic and causal inference: the works of Judea Pearl*, pages 373–392.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine Mcleavey, and Ilya Sutskever. 2023. Robust speech recognition via large-scale weak supervision. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 28492–28518. PMLR.

Senthooran Rajamanoharan, Tom Lieberum, Nicolas Sonnerat, Arthur Conmy, Vikrant Varma, János Kramár, and Neel Nanda. 2024. Jumping ahead: Improving reconstruction fidelity with jumprelu sparse autoencoders. *Preprint*, arXiv:2407.14435.

Lee Sharkey, Dan Braun, and Beren Millidge. 2023. Taking the temperature of transformer circuits. Accessed: 2024-08-18.

Aaquib Syed, Can Rager, and Arthur Conmy. 2023. Attribution patching outperforms automated circuit discovery. In *NeurIPS Workshop on Attributing Model Behavior at Scale*.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, Pouya Tafti, Léonard Hussenot, Pier Giuseppe Sessa, Aakanksha Chowdhery, Adam Roberts, Aditya Barua, Alex Botev, Alex Castro-Ros, Ambrose Slone, Amélie Héliou, Andrea Tacchetti, Anna Bulanova, Antonia Paterson, Beth Tsai, Bobak Shahriari, Charline Le Lan, Christopher A. Choquette-Choo, Clément Crepy, Daniel Cer, Daphne Ippolito, David Reid, Elena Buchatskaya, Eric Ni, Eric Noland, Geng Yan, George Tucker, George-Christian Muraru, Grigory Rozhdestvenskiy, Henryk Michalewski, Ian Tenney, Ivan Grishchenko, Jacob Austin, James Keeling, Jane Labanowski, Jean-Baptiste Lespiau, Jeff Stanway, Jenny Brennan, Jeremy Chen, Johan Ferret, Justin Chiu, Justin Mao-Jones, Katherine Lee, Kathy Yu, Katie Millican, Lars Lowe Sjoesund, Lisa Lee, Lucas Dixon, Machel Reid, Maciej Mikuła, Mateo Wirth, Michael Sharman, Nikolai Chinaev, Nithum Thain, Olivier Bachem, Oscar Chang, Oscar Wahltinez, Paige Bailey, Paul Michel, Petko Yotov, Rahma Chaabouni, Ramona Comanescu, Reena Jana, Rohan Anil, Ross McIlroy, Ruibo Liu, Ryan Mullins, Samuel L Smith, Sebastian Borgeaud, Sertan Girgin, Sholto Douglas, Shree Pandya, Siamak Shakeri, Soham De, Ted Klimenko, Tom Hennigan, Vlad Feinberg, Wojciech Stokowiec, Yu hui Chen, Zafarali Ahmed, Zhitao Gong, Tris Warkentin, Ludovic Peran, Minh Giang, Clément Farabet, Oriol Vinyals, Jeff Dean, Koray Kavukcuoglu, Demis Hassabis, Zoubin Ghahramani, Douglas Eck, Joelle Barral, Fernando Pereira, Eli

Collins, Armand Joulin, Noah Fiedel, Evan Senter, Alek Andreev, and Kathleen Kenealy. 2024a. Gemma: Open models based on gemini research and technology. *Preprint*, arXiv:2403.08295.

Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, Johan Ferret, Peter Liu, Pouya Tafti, Abe Friesen, Michelle Casbon, Sabela Ramos, Ravin Kumar, Charline Le Lan, Sammy Jerome, Anton Tsitsulin, Nino Vieillard, Piotr Stanczyk, Sertan Girgin, Nikola Momchev, Matt Hoffman, Shantanu Thakoor, Jean-Bastien Grill, Behnam Neyshabur, Olivier Bachem, Alanna Walton, Aliaksei Severyn, Alicia Parrish, Aliya Ahmad, Allen Hutchison, Alvin Abdagic, Amanda Carl, Amy Shen, Andy Brock, Andy Coenen, Anthony Laforge, Antonia Paterson, Ben Bastian, Bilal Piot, Bo Wu, Brandon Royal, Charlie Chen, Chintu Kumar, Chris Perry, Chris Welty, Christopher A. Choquette-Choo, Danila Sinopalnikov, David Weinberger, Dimple Vijaykumar, Dominika Rogozińska, Dustin Herbison, Elisa Bandy, Emma Wang, Eric Noland, Erica Moreira, Evan Senter, Evgenii Eltyshev, Francesco Visin, Gabriel Rasskin, Gary Wei, Glenn Cameron, Gus Martins, Hadi Hashemi, Hanna Klimczak-Plucińska, Harleen Batra, Harsh Dhand, Ivan Nardini, Jacinda Mein, Jack Zhou, James Svensson, Jeff Stanway, Jetha Chan, Jin Peng Zhou, Joana Carrasqueira, Joana Iljazi, Jocelyn Becker, Joe Fernandez, Joost van Amersfoort, Josh Gordon, Josh Lipschultz, Josh Newlan, Ju yeong Ji, Kareem Mohamed, Kartikeya Badola, Kat Black, Katie Millican, Keelin McDonell, Kelvin Nguyen, Kiranbir Sodhia, Kish Greene, Lars Lowe Sjoesund, Lauren Usui, Laurent Sifre, Lena Heuermann, Leticia Lago, Lilly McNealus, Livio Baldini Soares, Logan Kilpatrick, Lucas Dixon, Luciano Martins, Machel Reid, Manvinder Singh, Mark Iverson, Martin Görner, Mat Velloso, Mateo Wirth, Matt Davidow, Matt Miller, Matthew Rahtz, Matthew Watson, Meg Risdal, Mehran Kazemi, Michael Moynihan, Ming Zhang, Minsuk Kahng, Minwoo Park, Mofi Rahman, Mohit Khatwani, Natalie Dao, Nenshad Bardoliwalla, Nesh Devanathan, Neta Dumai, Nilay Chauhan, Oscar Wahltinez, Pankil Botarda, Parker Barnes, Paul Barham, Paul Michel, Pengchong Jin, Petko Georgiev, Phil Culliton, Pradeep Kuppala, Ramona Comanescu, Ramona Merhej, Reena Jana, Reza Ardeshir Rokni, Rishabh Agarwal, Ryan Mullins, Samaneh Saadat, Sara Mc Carthy, Sarah Perrin, Sébastien M. R. Arnold, Sebastian Krause, Shengyang Dai, Shruti Garg, Shruti Sheth, Sue Ronstrom, Susan Chan, Timothy Jordan, Ting Yu, Tom Eccles, Tom Hennigan, Tomas Kocisky, Tulsee Doshi, Vihan Jain, Vikas Yadav, Vilobh Meshram, Vishal Dharmadhikari, Warren Barkley, Wei Wei, Wenming Ye, Woohyun Han, Woosuk Kwon, Xiang Xu, Zhe Shen, Zhitao Gong, Zichuan Wei, Victor Cotruta, Phoebe Kirk, Anand Rao, Minh Giang, Ludovic Peran, Tris Warkentin, Eli Collins, Joelle Barral, Zoubin Ghahramani, Raia Hadsell, D. Sculley, Jeanine Banks, Anca Dragan, Slav Petrov, Oriol Vinyals, Jeff Dean, Demis Hassabis, Koray Kavukcuoglu, Clement Farabet, Elena Buchatskaya, Sebastian Borgeaud, Noah Fiedel, Armand Joulin, Kathleen Kenealy, Robert Dadashi, and Alek Andreev. 2024b. Gemma 2: Improving open language models at a practical size. *Preprint*, arXiv:2408.00118.

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, Shan Carter, Chris Olah, and Tom Henighan. 2024. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. *Transformer Circuits Thread*.

Ethan Waisberg, Joshua Ong, Mouayad Masalkhi, Sharif Amit Kamran, Nasif Zaman, Prithul Sarker, Andrew G Lee, and Alireza Tavakkoli. 2023. Gpt-4 and ophthalmology operative notes. *Annals of Biomedical Engineering*, 51(11):2353–2355.

Kevin Ro Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. 2023. Interpretability in the wild: a circuit for indirect object identification in GPT-2 small. In *The Eleventh International Conference on Learning Representations*.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, Quan Vuong, Vincent Vanhoucke, Huong Tran, Radu Soricut, Anikait Singh, Jaspiar Singh, Pierre Sermanet, Pannag R. Sanketi, Grecia Salazar, Michael S. Ryoo, Krista Reymann, Kanishka Rao, Karl Pertsch, Igor Mordatch, Henryk Michalewski, Yao Lu, Sergey Levine, Lisa Lee, Tsang-Wei Edward Lee, Isabel Leal, Yuheng Kuang, Dmitry Kalashnikov, Ryan Julian, Nikhil J. Joshi, Alex Irpan, Brian Ichter, Jasmine Hsu, Alexander Herzog, Karol Hausman, Keerthana Gopalakrishnan, Chuyuan Fu, Pete Florence, Chelsea Finn, Kumar Avinava Dubey, Danny Driess, Tianli Ding, Krzysztof Marcin Choromanski, Xi Chen, Yevgen Chebotar, Justice Carbajal, Noah Brown, Anthony Brohan, Montserrat Gonzalez Arenas, and Kehang Han. 2023. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pages 2165–2183. PMLR.

A IE estimation through Attribution Patching

In Equation 7 we reported the Indirect Effect (IE) (Pearl, 2022), which measures the importance of a feature with respect to a generic downstream task $\mathcal{T}$. To reduce the computational burden of estimating the IE with a single forward pass per feature, we employed Attribution Patching (AtP) (Nanda, 2023; Syed et al., 2023). AtP employs a first-order Taylor expansion

$$\hat{\text{IE}}_{\text{AtP}}(m; \mathbf{f}; a_c, a_w) = \nabla_{\mathbf{f}} m\big|_{\mathbf{f}=\mathbf{f}_a}(\mathbf{f}_w - \mathbf{f}_a) \quad (10)$$

which estimates Equation 7 for every $\mathbf{f}$ in two forward passes and a single backward pass.

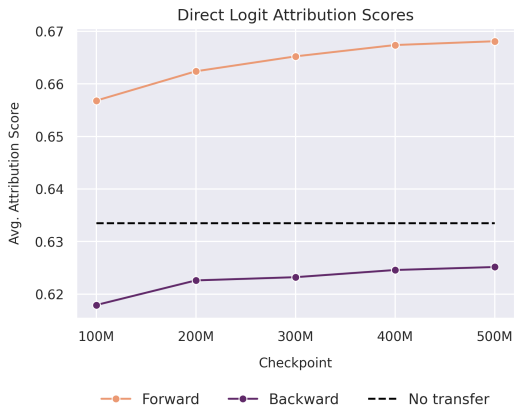

Figure 8: Direct Logit Attribution Scores averaged across layers for every tested checkpoint compared to the "No Transfer" baseline, i.e. the DLA scores obtained by $\text{SAE}_i(\mathbf{x}_i), \forall i = 0, ..., 11$.

B Direct Logit Attribution

We also report the Direct Logit Attribution (DLA) between forward $\text{SAE}_{i \leftarrow i-1}$ and backward $\text{SAE}_{i \leftarrow i+1}$ transfer SAEs. Introduced by Bricken et al. (2023), DLA assesses the direct effect of a feature on the next-token distribution, providing insights into the causal role of features. The attribution score is computed as follows:

$$\text{attr}_i(x; a_c; a_w) = \text{f}_i \, \mathbf{v}_i \cdot \nabla_x \mathcal{L}(a_c, a_w) \quad (11)$$

where x is a given prompt and $\nabla_x \mathcal{L}$ is the gradient of the logit difference between two contrastive answer tokens a_c, a_w (E.g., x = "The square root of 9 is", $a_c = 3$, and $a_w = 2$). We report the feature averaged DLA computed on a custom dataset comprising 64 handcrafted prompts in the form of $\{(x, a_c, a_w)_i\}$. Figure 8 displays the layer-averaged DLA scores for each tested checkpoint. The plot reveals that forward transfer SAEs consistently achieves higher scores than the baseline, while backward transfer SAEs consistently scores lower. This outcome contrasts with the reconstruction metrics, where the backward technique consistently outperformed the forward approach. A detailed per-layer DLA scores plot is reported in Figure 22.

B.1 Compute Efficiency

This work proposes a novel method leveraging transfer learning to significantly reduce computational costs in training SAEs in the context of LLMs. We demonstrate that both Fwd-SAE $\text{SAE}_{i \leftarrow i-1}$ and Bwd-SAE $\text{SAE}_{i \leftarrow i+1}$, trained with our fine-tuning strategy, are both valid alternatives to the standard layer-by-layer training of SAE_i, in terms of both reconstruction quality of the learned representation and performance on downstream tasks. In practice, our approach consists of the following steps:

1. Train a SAE_i on alternate layers, depending on the transfer direction. For Forward transfer $i \in \{0, 2, 4, ..., L\}$, while for Backward transfer $i \in \{1, 3, 5, ..., L-1\}$.

2. Initialize the current SAE_i by either $\text{SAE}_{i \leftarrow i-1}$ for forward transfer or $\text{SAE}_{i \leftarrow i+1}$ for backward transfer.

3. Apply transfer learning by training the remaining SAEs and stop when some criteria are matched (e.g., when the loss converges to a specific value or when a computational budget has been reached).

Empirical results demonstrate substantial efficiency gains. In our experiments with a 12-layer Pythia-160M (Biderman et al., 2023) model, we observed a performance increase after fine-tuning on 10% of the training data (Figure 3 and Figure 4), with performance increasing over time. Extrapolating these findings, we can compute empirical lower and upper bounds on the training efficiency. Given a model with L (in our particular case $L = 12$) layers and a training set consisting of 1B tokens, we have:

- **Baseline training:** Train one SAE_i $\forall i \in \{1, ..., 12\}$ for 1B tokens: 12B tokens

- **Forward/Backward transfer - 10% of data**:

  - Train one SAE_i for half of the layers for 1B tokens: 6B tokens
  - Fine-tune the remaining $\text{SAE}_{i \leftarrow i-1}$ or $\text{SAE}_{i \leftarrow i+1}$ for 100M tokens: 0.6B tokens
  - **Total**: 6.6B tokens

- **Forward/Backward transfer - 50% of data**:
  - Train one SAE_i for half of the layers for 1B tokens: 6B tokens
  - Fine-tune the remaining $\text{SAE}_{i \leftarrow i-1}$ or $\text{SAE}_{i \leftarrow i+1}$ for 500M tokens: 3B tokens
  - **Total**: 9B tokens

- **Computational savings:**
  - **Lower bound** Forward/Backward transfer: 12B - 6.6B = 5.4B tokens
  - **Upper bound** Forward/Backward transfer: 12B - 9B = 3B tokens

- **Relative reduction in compute cost:**
  - **Lower bound** Forward/Backward transfer: $\frac{5.4\text{B}}{12\text{B}} \times 100\% = 45\%$
  - **Upper bound** Forward/Backward transfer: $\frac{9\text{B}}{12\text{B}} \times 100\% = 25\%$

Our analysis indicates that the proposed transfer learning approach can reduce compute costs by 25% to 45% for forward and backward transfer when fine-tuned for 50% and 10% of the training data respectively, improving efficiency and reducing costs by a great margin, while maintaining both reconstruction quality and performance on downstream tasks.

C Additional plots and tables

Hyperparameter	Value
c	8
λ	1.0
Hook name	resid-post
Batch size	4096
Adam (β_1, β_2)	$(0, 0.999)$
lr (Train)	3e-5
lr (Fine-tuning)	1e-5
lr scheduler	constant
lr deacy steps	20% of the training steps
l1 warm-up steps	5% of the training steps
# tokens (Train)	1B
# tokens (Fine-tuning)	500M
Checkpoint freq.	100M

Table 2: Training and fine-tuning hyperparameters

Checkpoint	i										
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
100M	0.962	0.960	0.983	0.920	0.865	0.439	0.955	0.948	0.858	0.944	1.003
200M	0.968	0.968	0.996	0.933	0.873	0.459	0.970	0.956	0.894	0.965	1.005
300M	0.969	0.971	1.000	0.941	0.877	0.475	0.981	0.960	0.911	0.972	1.005
400M	0.971	0.974	1.003	0.944	0.879	0.479	0.988	0.963	0.921	0.978	1.006
500M	0.972	0.975	1.005	0.946	0.881	0.488	0.991	0.964	0.929	0.981	1.006

Table 3: Normalized CE-Loss Scores $\overline{\text{CES}}_{i,i-1}$ (Eq. 9) of the Fwd-SAE at different checkpoints. On $i = 6$, the Normalized CE-Loss Score increases over time even though it starts with a lower value w.r.t. the other checkpoints. From Figure 9 we note how the CE-Loss Score of $\text{SAE}_5(\mathbf{x}_6)$ and $\text{SAE}_{6\leftarrow 5}(\mathbf{x}_6)$ are nearly identical to the obtained by $\text{SAE}_6(\mathbf{x}_6)$, thus the increment given by the fine-tuning over the baseline $\text{SAE}_5(\mathbf{x}_6)$, captured by the Normalized CE-Loss Score in Eq. 9, is minimal and resulting in a lower value.

Checkpoint	i										
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
100M	0.988	0.927	0.964	1.052	0.803	0.375	0.801	1.044	0.920	1.005	0.939
200M	0.990	0.939	0.969	1.076	0.812	0.396	0.805	1.047	0.912	1.001	0.953
300M	0.991	0.945	0.972	1.084	0.823	0.412	0.808	1.049	0.913	0.999	0.965
400M	0.995	0.951	0.975	1.098	0.827	0.420	0.811	1.052	0.912	0.997	0.972
500M	0.997	0.951	0.975	1.098	0.827	0.425	0.814	1.056	0.913	0.998	0.976

Table 4: Normalized CE-Loss Scores $\overline{\text{CES}}_{i,i+1}$ of the Bwd-SAE at different checkpoints. On $i = 5$, the Normalized CE-Loss Score increases over time even though it starts with a lower value w.r.t. the other checkpoints. From Figure 9 we note how the CE-Loss Score of $\text{SAE}_6(\mathbf{x}_5)$ and $\text{SAE}_{5\leftarrow 6}(\mathbf{x}_5)$ are nearly identical to the obtained by $\text{SAE}_5(\mathbf{x}_5)$, thus the increment given by the fine-tuning over the baseline $\text{SAE}_6(\mathbf{x}_5)$, captured by the Normalized CE-Loss Score in Eq. 9, is minimal and resulting in a lower value.

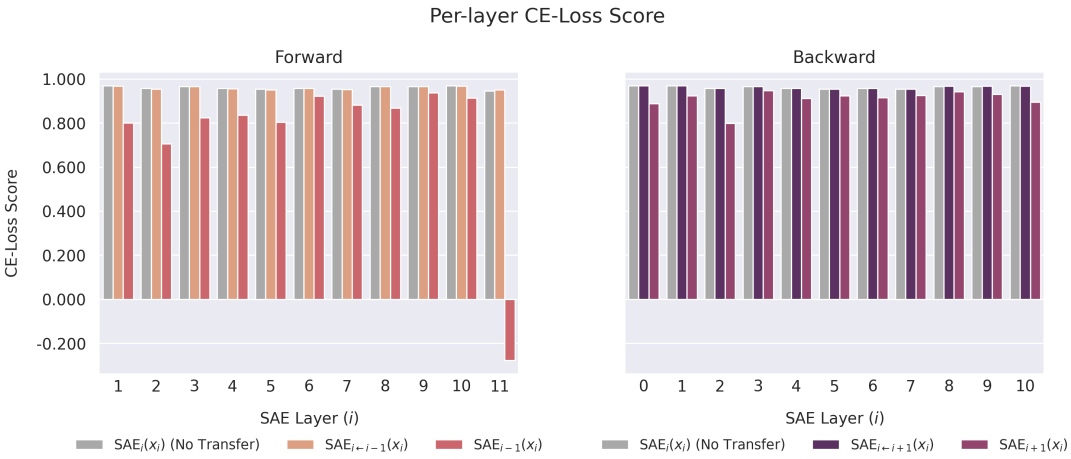

Figure 9: Detailed per-layer CE-Loss Score at the final checkpoint (500M). $\text{SAE}_{i-1}(\mathbf{x}_i)$ and $\text{SAE}_{i+1}(\mathbf{x}_i)$ are the baselines for the Fwd-SAE and Bwd-SAE respectively.

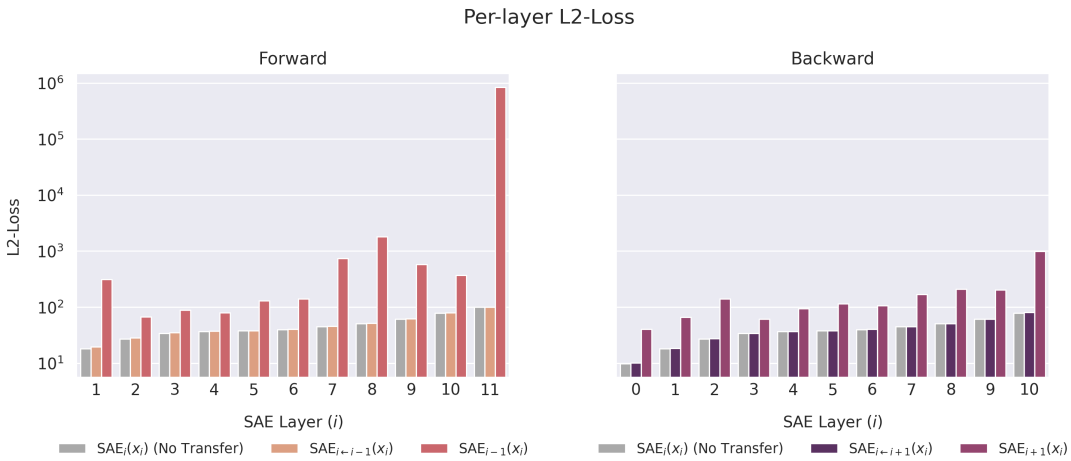

Figure 10: Detailed per-layer $L2$-Loss at the final checkpoint (500M). $\text{SAE}_{i-1}(\mathbf{x}_i)$ and $\text{SAE}_{i+1}(\mathbf{x}_i)$ are the baselines for the Fwd-SAE and Bwd-SAE respectively. The y-axis is on a logarithmic scale.

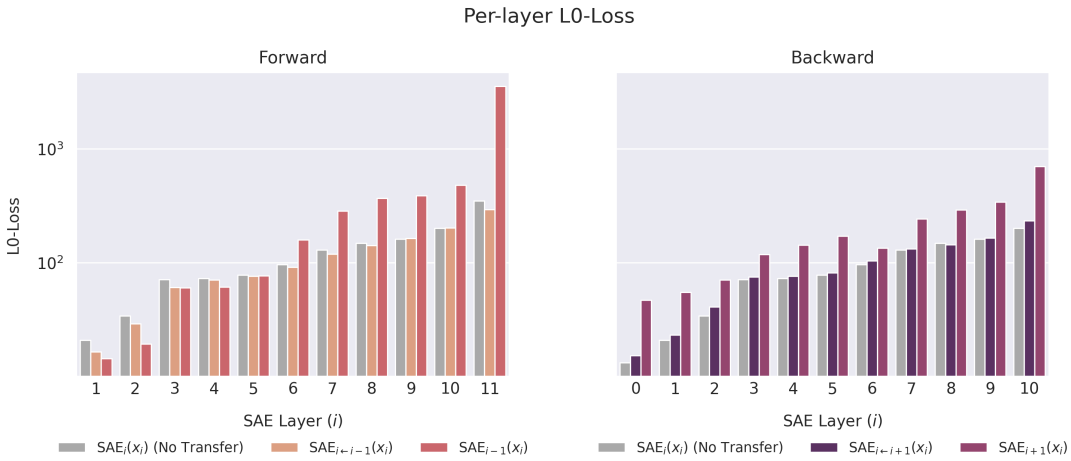

Figure 11: Detailed per-layer $L0$-Loss at the final checkpoint (500M). $\text{SAE}_{i-1}(\mathbf{x}_i)$ and $\text{SAE}_{i+1}(\mathbf{x}_i)$ are the baselines for the Fwd-SAE and Bwd-SAE respectively. The y-axis is on a logarithmic scale.

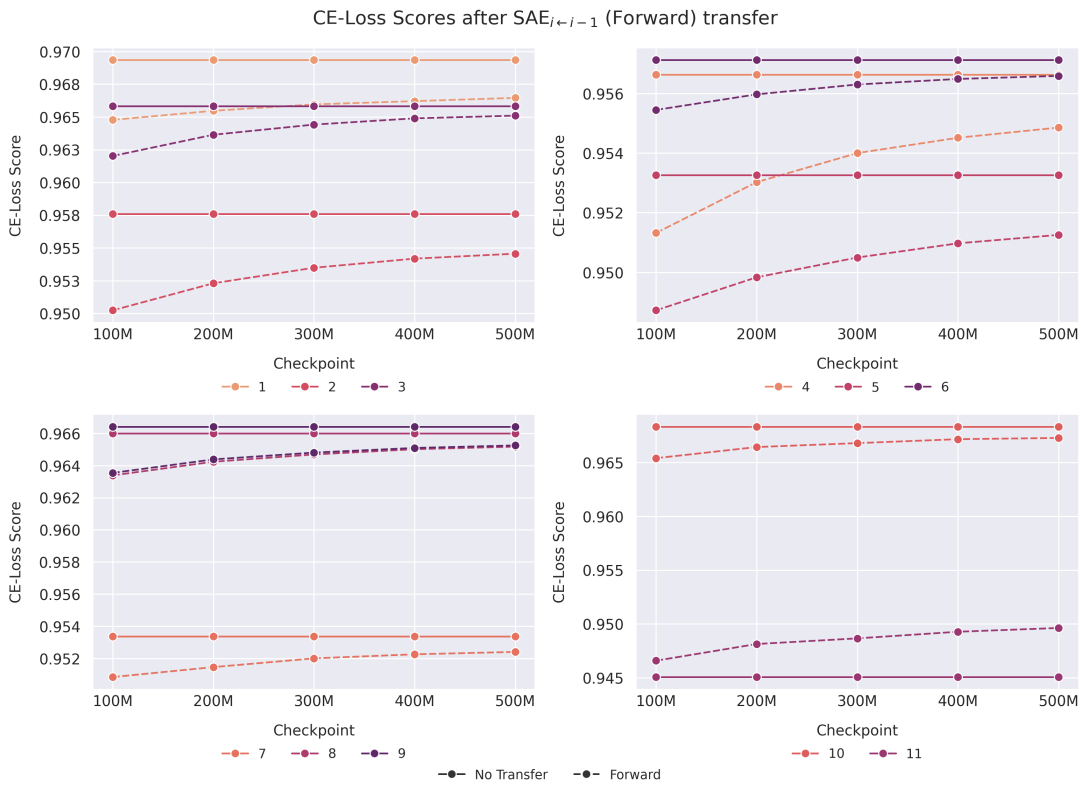

Figure 12: Detailed per-layer CE-Loss Score over time (Checkpoint) after Forward Transfer.

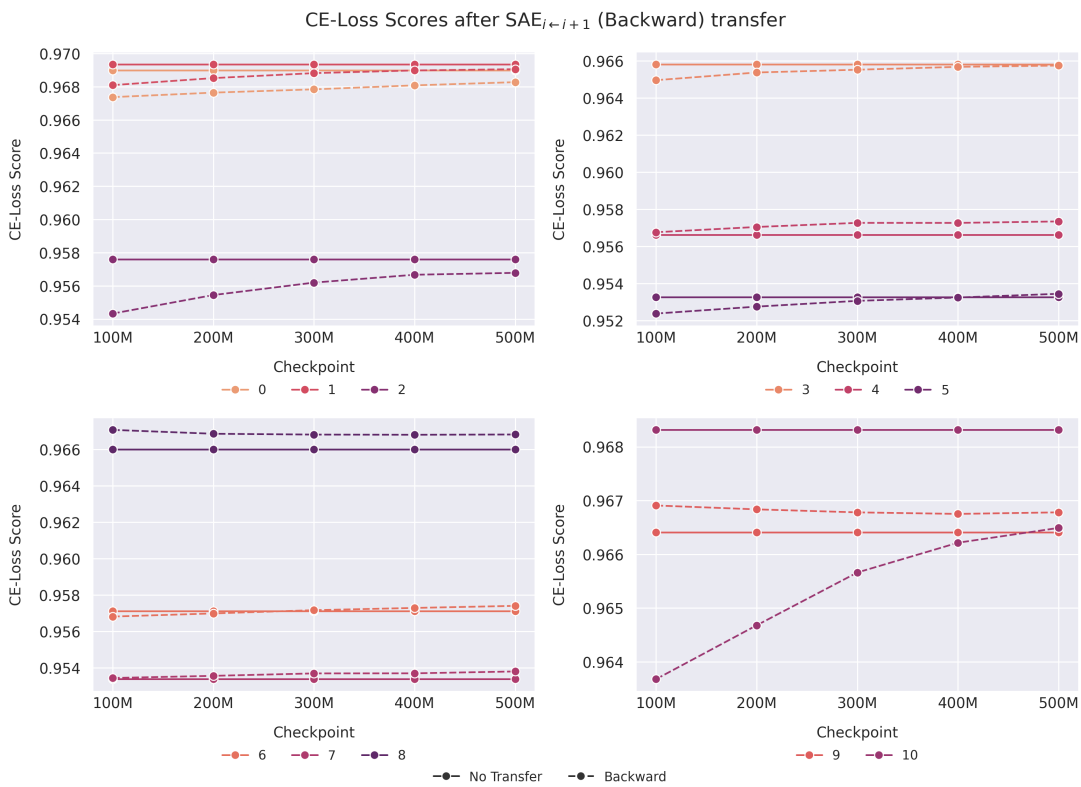

Figure 13: Detailed per-layer CE-Loss Score over time (Checkpoint) after Backward Transfer.

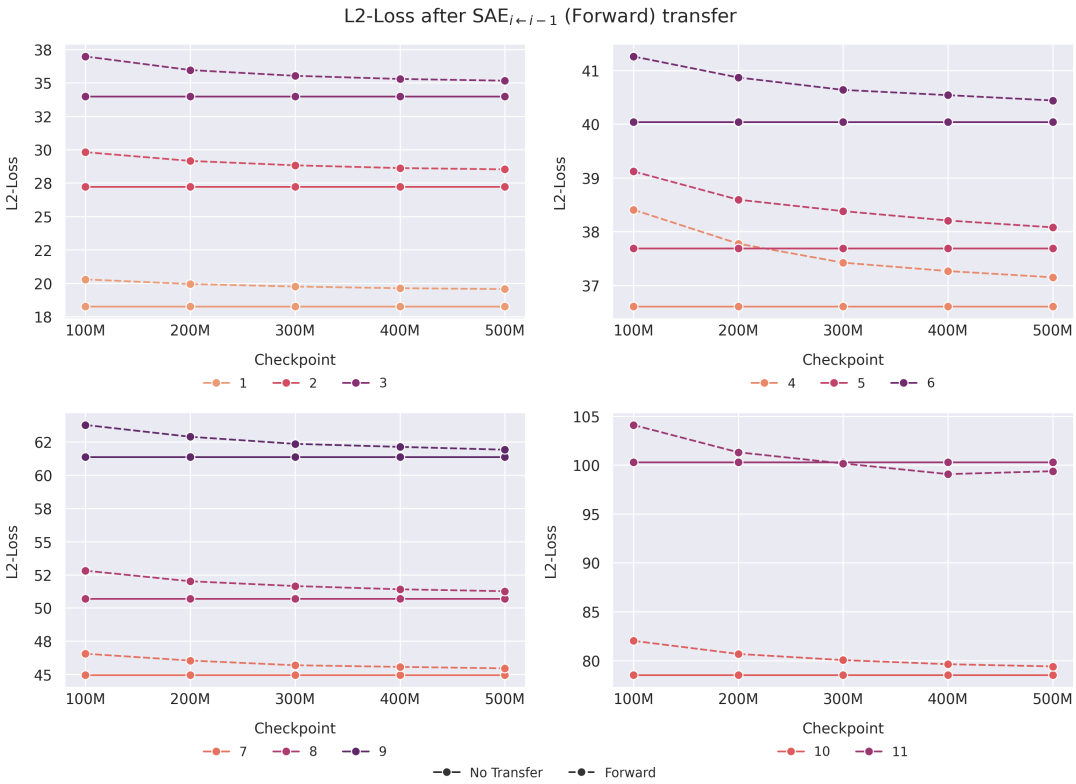

Figure 14: Detailed per-layer $L2$-Loss over time (Checkpoint) after Forward Transfer.

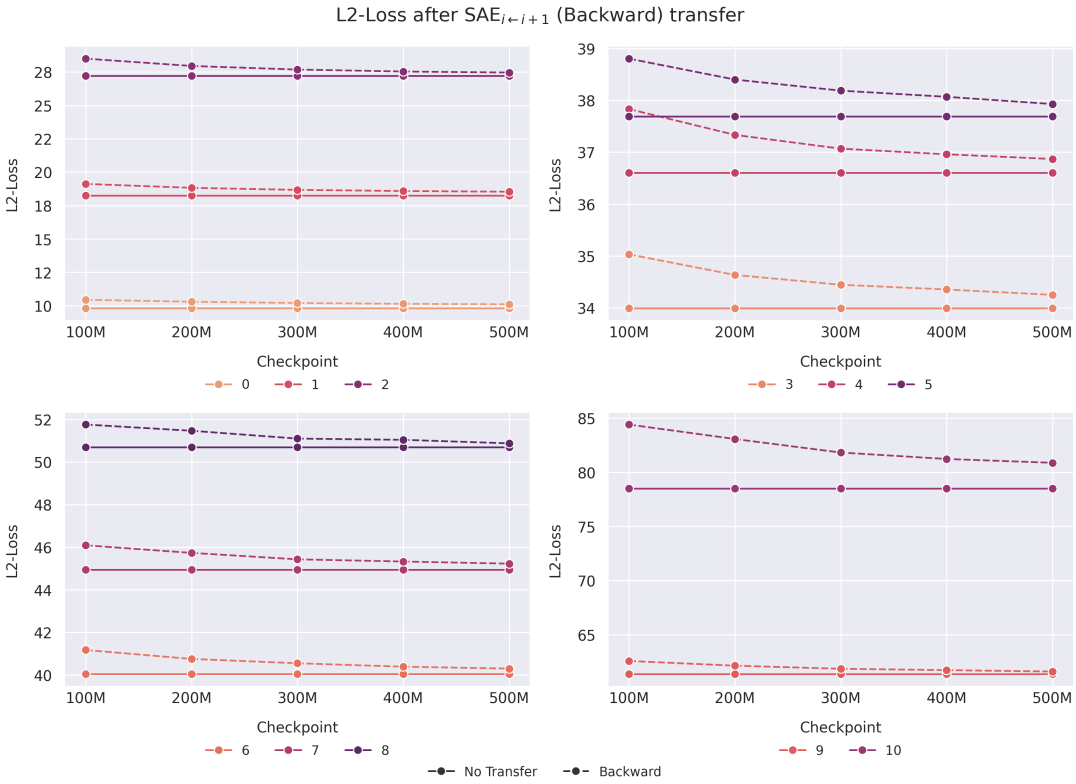

Figure 15: Detailed per-layer $L2$-Loss over time (Checkpoint) after Backward Transfer.

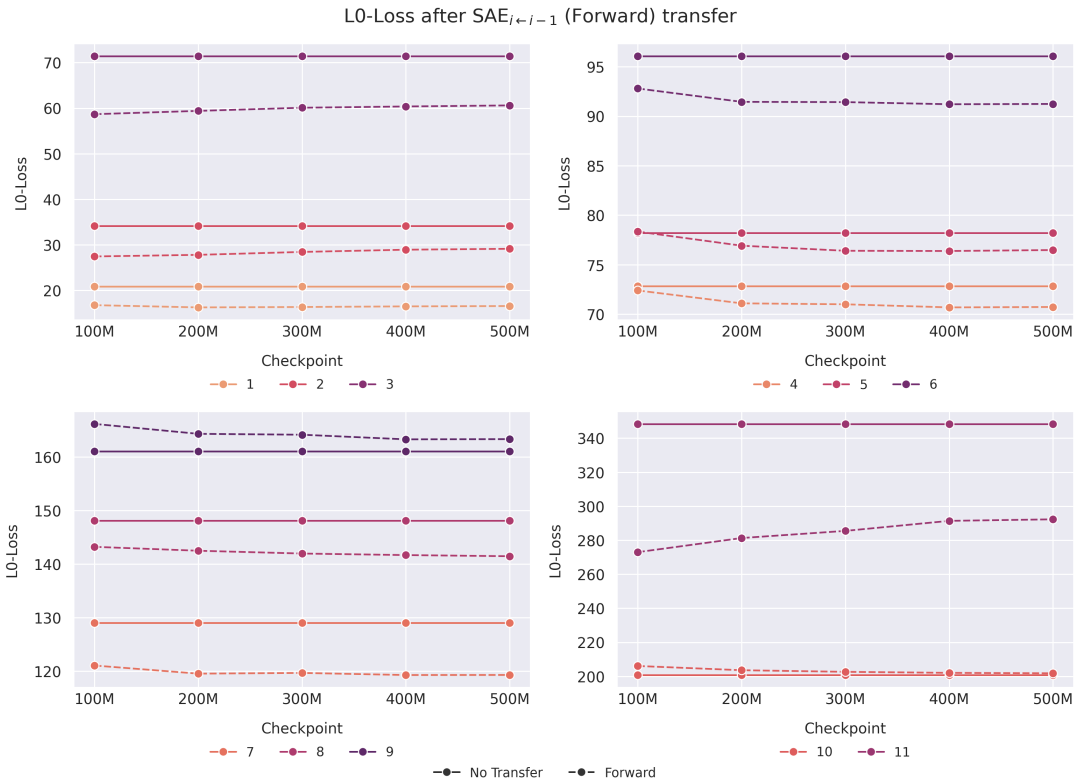

Figure 16: Detailed per-layer $L0$-Loss over time (Checkpoint) after Forward Transfer.

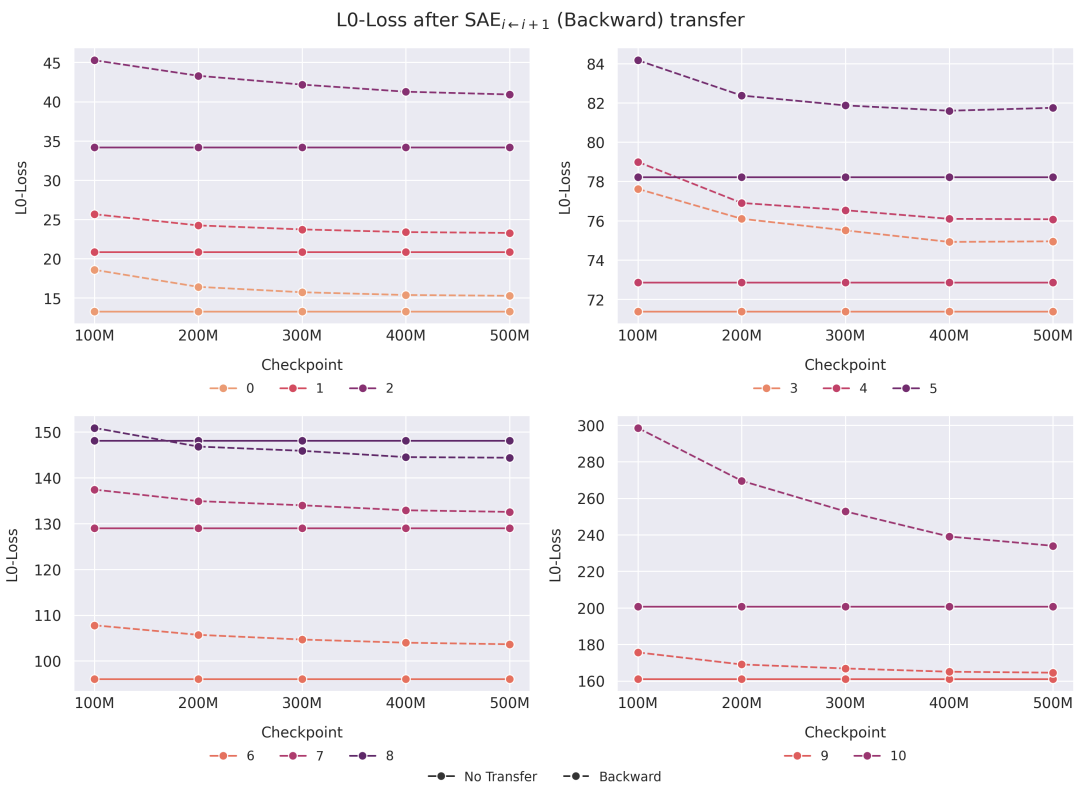

Figure 17: Detailed per-layer $L0$-Loss over time (Checkpoint) after Backward Transfer.

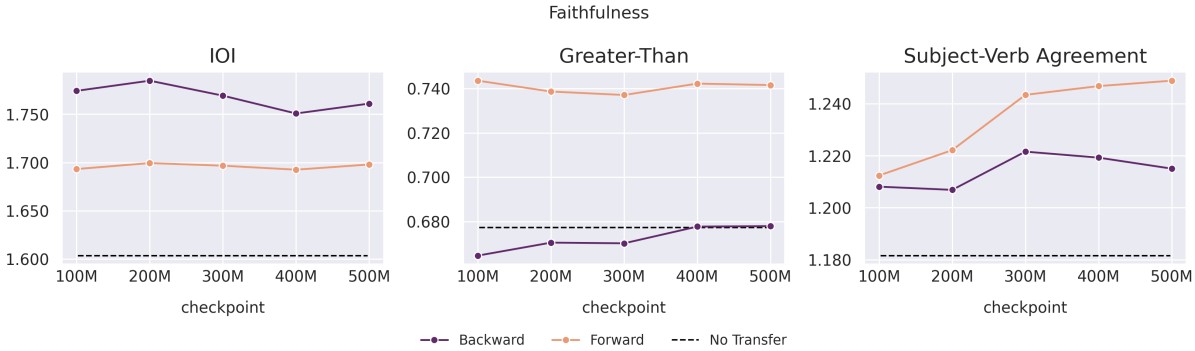

Figure 18: Faithfulness over time (Checkpoint) averaged by layer and N for the three downstream tasks.

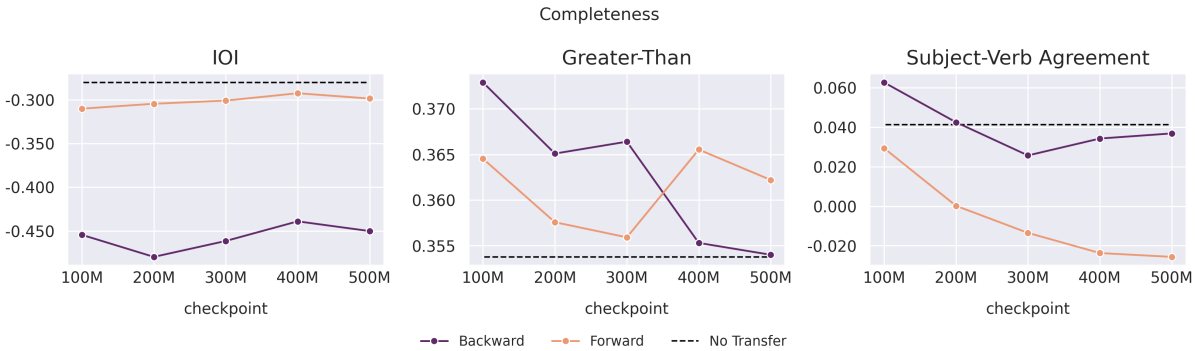

Figure 19: Completeness over time (Checkpoint) averaged by layer and N for the three downstream tasks.

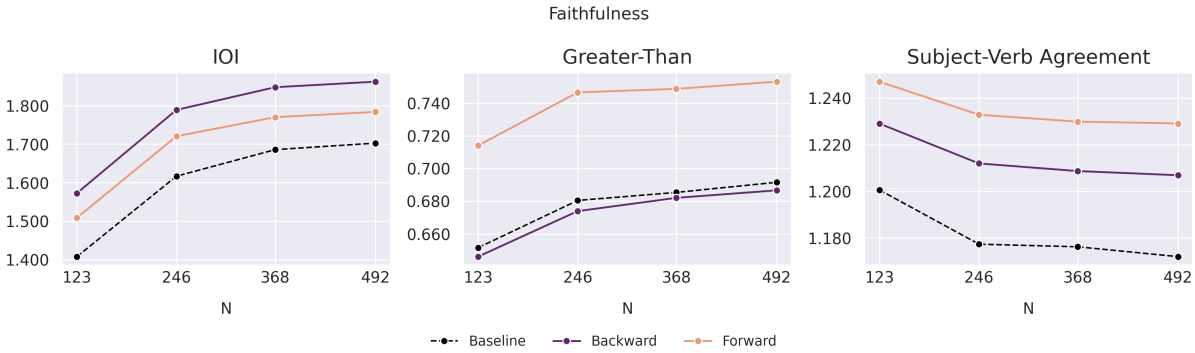

Figure 20: Faithfulness over N averaged by layer and time (Checkpoints) for the three downstream tasks.

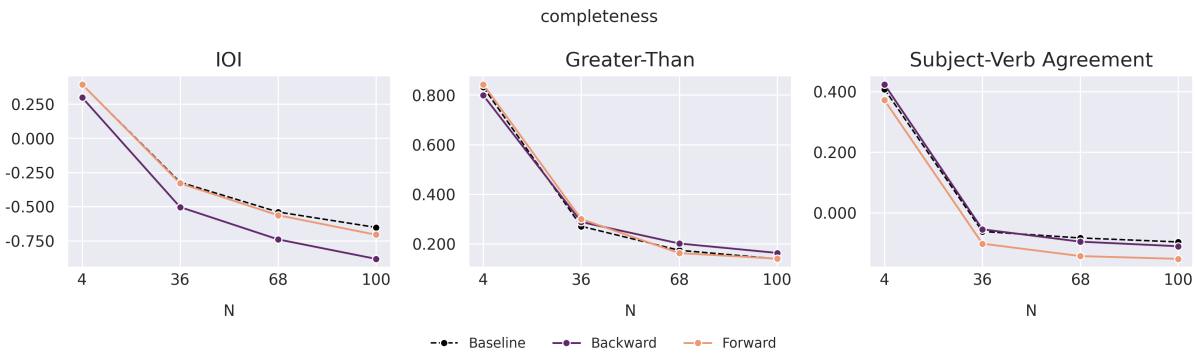

Figure 21: Completeness over N averaged by layer and time (Checkpoints) for the three downstream tasks.

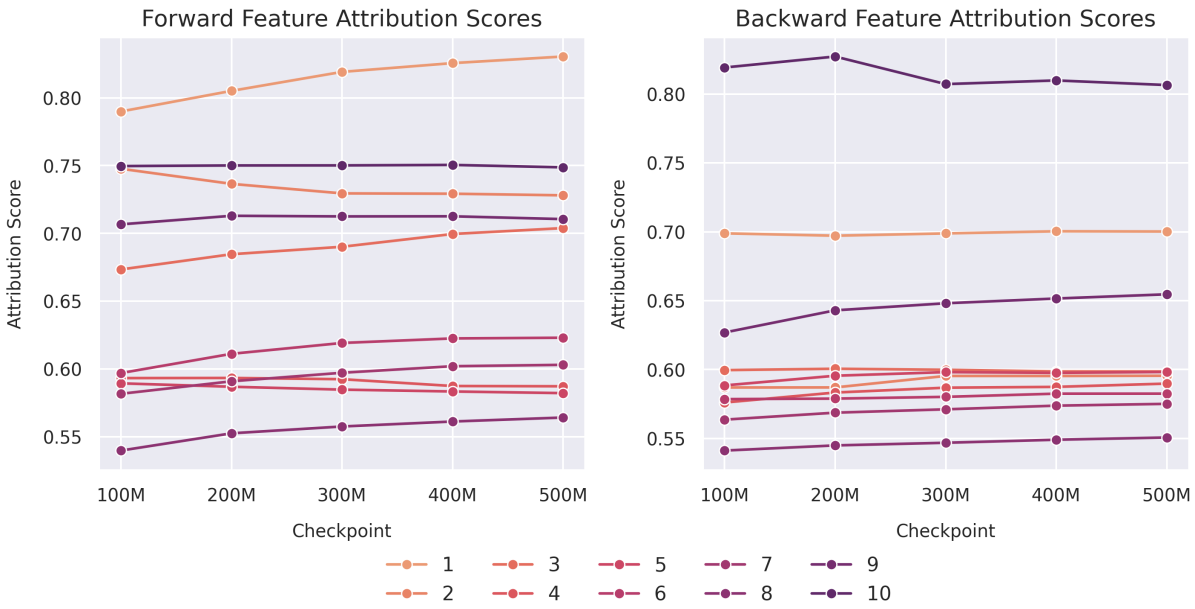

Figure 22: Detailed per-layer feature averaged Logits Attribution scores over time (Checkpoint), as defined in Equation 11.

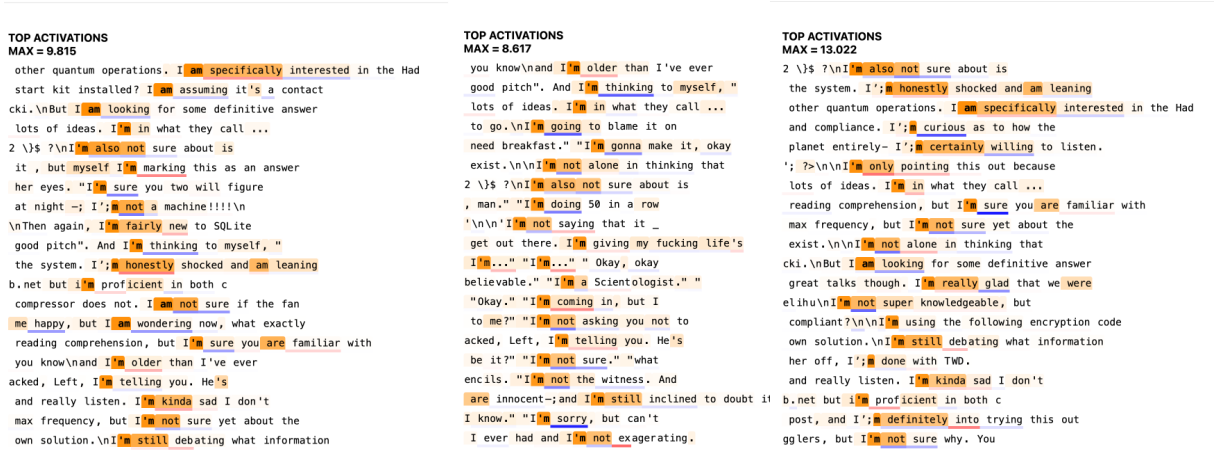

Figure 23: Comparison of top activations of feature 949 across layer 8 SAE and two transfer SAEs pre-trained on the former. SAE_8 (Left), $\text{SAE}_{7\leftarrow 8}$ (Middle), $\text{SAE}_{9\leftarrow 8}$ (Right). Evidence of feature transfer across three layers.