Robustly identifying concepts introduced during chat fine-tuning using crosscoders

Julian Minder[*,ə] **Clément Dumas**[*,†,‡]
Caden Juang[ð] **Bilal Chughtai** **Neel Nanda**
[ə]EPFL [†]Ecole Normale Supérieure Paris-Saclay [‡]Université Paris-Saclay
[ð]Northeastern University
julian.minder@epfl.ch, clement.dumas@ens-paris-saclay.fr

Abstract

Model diffing studies the *change* in a model's representations or internal algorithms as a result of fine-tuning. Focusing on such differences may offer a promising lens into a range of model behaviors of interest, and might be significantly easier than attempts to reverse engineer and interpret the entire model. The recently introduced crosscoder (Lindsey et al., 2024) enables such analysis by learning a shared dictionary of interpretable concepts represented as latent directions in both models, allowing us to track how concepts shift or emerge during fine-tuning. In this work, we identify two issues with the crosscoder training objective, which we term Complete Shrinkage and Latent Decoupling – that can misattribute concepts as unique to the fine-tuned model via prior naive specificity metrics, when they really exist in both models. We develop a technique we call Latent Scaling to address these issues – which more accurately measures each latent's presence across both base and chat models, allowing us to identify which concepts are more genuinely specific to the fine-tuned model. The concepts we identify are both more causally effective in controlling chat-model specific behavior and have higher qualitative latent interpretability. Latents with high "specificity" represent concepts such as "knowledge gap identifier" or "self-identity" and primarily activate on tokens that structure model conversations. Overall, our work improves the crosscoder-based methodology for model diffing and provides concrete insights into how chat tuning modifies the behavior of language models.

1 Introduction

Classically, the goal of mechanistic interpretability (Sharkey et al., 2025; Mueller et al., 2024; Ferrando et al., 2024; Elhage et al., 2021; Olah et al., 2020) research has been to understand either an entire model (Huben et al., 2024; Elhage et al., 2022), or to understand specific *circuits*, or algorithms, that are implemented by the model to solve particular tasks (Wang et al., 2023a). This is akin to trying to understand the entire source code of a running computer program, and is challenging. *Model diffing* is a relatively nascent approach that instead attempts to detect what has *changed* in a model as a result of fine-tuning. Given the relatively small compute used for present-day fine-tuning compared to pre-training, we expect the changes introduced to be limited in scope – perhaps akin to a pull request on a large code repository.

[*]Equal contribution. Order randomized.

Pretraining teaches the model general world knowledge, generic circuitry and skills. These are broadly useful in a variety of settings. Fine-tuning has little reason to change most of this cognition. It seems likely the fine-tuned model will share many representations with the base model, and only specific aspects will change. For instance, the model's persona, chat specific skills that help it follow instructions and reply to users, and other task specific skills more broadly. This argument suggests that the model diffing approach to mechanistic interpretability might be comparatively *easier* than trying to understand the full model (though see Section XX for arguments against).

Model diffing might also be incredibly useful. The process of fine-tuning a model is what makes it *useful* as a tool or agent. Better understanding the mechanisms that give reasoning models (DeepSeek-AI et al., 2025; OpenAI et al., 2024) heightened capabilities as compared to base or chat models might allow us to debug their failures and improve them. Fine-tuning also often introduces a number of problematic behaviours, for instance sycophancy (Sharma et al., 2023). Future AI safety and alignment concerns (Greenblatt et al., 2024; Meinke et al., 2025) may emerge specifically in fine-tuned models. For example, long-horizon RL could incentivize models to exploit reward signals and act deceptively, building on deception concepts already learned during pretraining. It's possible model diffing will be sufficient to allow us to detect this.

julian
Clement any other that come to mind?

Prior model diffing research has investigated how models change during fine-tuning (Lindsey et al., 2024; Bricken et al., 2024; Prakash et al., 2024; Lee et al., 2024; Jain et al., 2024; Khayatan et al., 2025; Thasarathan et al., 2025; Wu et al., 2024; Mosbach, 2023; Merchant et al., 2020; Hao et al., 2020; Kovaleva et al., 2019). While these studies have hypothesized that fine-tuning primarily shifts and repurposes existing capabilities rather than developing entirely new ones, conclusive evidence for this claim remains elusive. Model diffing remains a nascent field that lacks established consensus and mature analytical tools. Much prior work has leveraged ad-hoc techniques for understanding how models change in narrow ways (e.g. studying how a particular circuit, algorithm, or representation changes) (Prakash et al., 2024; Lee et al., 2024; Wu et al., 2024; Hao et al., 2020; Kovaleva et al., 2019), or have been on toy models (Jain et al., 2024). It is unclear whether many prior approaches would scale to understanding the kinds of fine-tuning large models actually undergo.

Recently, Lindsey et al. (2024) introduced a new tool for model diffing, the **crosscoder**, which may overcome the issues discussed above. Crosscoders build on the popular sparse autoencoder (SAE) (Huben et al., 2024; Bricken et al., 2023; Yun et al., 2021), which has shown promise for interpreting a model's representations by decomposing activations into a sum of sparsely activating dictionary elements. There are many variants of crosscoders; the variant we are concerned with in this paper are concatenating the activations of the base and fine-tuned model residual streams and train a shared dictionary across this activation stack. Thus, for each dictionary element (aka "latent", corresponding to one concept), the crosscoder learns a pair of latent directions - one corresponding to the base model and one to the fine-tuned model. Crosscoders can thus potentially identify which latents are novel to the fine-tuned model, which are novel to the base-model, and which are shared. We term these sets chat-only, shared, and base-only respectively.

In this work, we directly build on Lindsey et al. (2024). We critically examine the crosscoder, and its efficacy for model diffing. Our contributions are as follows:

1. We replicate the model diffing results from Lindsey et al. (2024) on the Gemma-2-2b and Gemma-2-2b-it pair of models (Section 3.1).
2. We identify two theoretical limitations of the crosscoder training objective, that may lead to falsely identified chat-only latents (Section 3.2).
   (a) Complete Shrinkage: The sparsity loss can force base latents to zero even when they contribute to base model reconstruction, particularly when a latent is more important for the chat model but still relevant for the base model.

   (b) Latent Decoupling: The crosscoder may represent a shared concept using a *chat-only* latent when it is actually encoded by a different combination of latents in the base model, as the crosscoder's sparsity loss treats both representations as equivalent.
3. We show the above issues arise in practice, and develop an approach we call *Latent Scaling* to detect and filter spurious chat-only latents, inspired by Wright & Sharkey (2024) (Section 3.2.2).
4. We show features identified as chat-only via latent scaling are more interpretable (Section 3.2.4) and more causally related to chat behavior (Section 3.2.5) than those identified as chat-only via the techniques used in Lindsey et al. (2024).
5. We characterize the role of chat-template tokens' role in chat behavior (Section 3.2.4) using our understanding of chat-specific concepts.

2 Background

We consider a crosscoder architecture (Lindsey et al., 2024) with two separate encoders and decoders, one corresponding to the base model and one to the chat model. Let x be the model input and $\mathbf{h}^{\text{base}}(x), \mathbf{h}^{\text{chat}}(x) \in \mathbb{R}^d$ denote the activations at a given layer. For a dictionary of size D, the latent activation of the j^{th} latent $f_j(x), j \in \{1, \ldots, D\}$ is computed as

$$f_j(x) = \text{ReLU}\left(\mathbf{e}_j^{\text{base}}\mathbf{h}^{\text{base}}(x) + \mathbf{e}_j^{\text{chat}}\mathbf{h}^{\text{chat}}(x) + b_j^{\text{enc}}\right) \tag{1}$$

where $\mathbf{e}_j^{\text{base}}, \mathbf{e}_j^{\text{chat}} \in \mathbb{R}^d$ are the corresponding encoder vectors and $b_j^{\text{enc}} \in \mathbb{R}$ is the encoder bias. The reconstructed activations for both models are then defined as:

$$\widetilde{\mathbf{h}}^{\text{base}}(x) = \sum_j f_j(x)\, \mathbf{d}_j^{\text{base}} + \mathbf{b}^{\text{dec,base}}$$
$$\widetilde{\mathbf{h}}^{\text{chat}}(x) = \sum_j f_j(x)\, \mathbf{d}_j^{\text{chat}} + \mathbf{b}^{\text{dec,chat}}$$

where $\mathbf{d}_j^{\text{base}}, \mathbf{d}_j^{\text{chat}} \in \mathbb{R}^d$ are the j^{th} decoder latents and $\mathbf{b}^{\text{dec,base}}, \mathbf{b}^{\text{dec,chat}} \in \mathbb{R}^d$ are the decoder biases.

The training loss for the crosscoder is a modified L1 SAE objective:

$$\begin{aligned} L(x) = &\frac{1}{2}\|\mathbf{h}^{\text{base}}(x) - \widetilde{\mathbf{h}}^{\text{base}}(x)\|_2 \\ &+ \frac{1}{2}\|\mathbf{h}^{\text{chat}}(x) - \widetilde{\mathbf{h}}^{\text{chat}}(x)\|_2 \\ &+ \mu \sum_j f_j(x)\left(\|\mathbf{d}_j^{\text{base}}\|_2 + \|\mathbf{d}_j^{\text{chat}}\|_2\right) \end{aligned} \tag{2}$$

with μ controlling the weight of the sparsity regularization term.[1] We define the reconstruction errors for the base and chat models as: $\boldsymbol{\varepsilon}^{\text{base}} = \mathbf{h}^{\text{base}} - \widetilde{\mathbf{h}}^{\text{base}}$ and $\boldsymbol{\varepsilon}^{\text{chat}} = \mathbf{h}^{\text{chat}} - \widetilde{\mathbf{h}}^{\text{chat}}$.

[1] While similar to training an SAE on concatenated activations, the crosscoder's sparsity loss uniquely promotes decoder norm differences (see Appendix A.1).

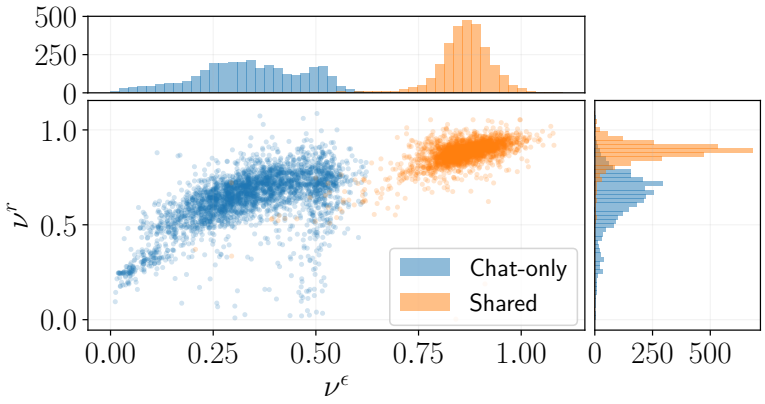


Figure 1: We measure the extent to which *chat-only* latents are affected by the two issues described in Section 3.2.1. Each datapoint represents a single crosscoder latent. The y-axis shows the reconstruction ratio ν^r, where high values with significant overlap with the *shared* distribution indicate *Latent Decoupling* – the *chat-only* latent carries information that is redundantly encoded by other *shared* or *base-only* latents. The x-axis displays the error ratio ν^ε, where high values indicate *Complete Shrinkage* – the loss forced this latent to have zero norm in the base decoder, although it is useful for the base model. Low values on both metrics indicate *true* chat-only latents. Many *chat-only* latents appear to be misidentified according to these metrics.

3 Results

3.1 Replicating Lindsey et al. (2024)'s model diffing results

We replicate the model diffing experiments by Lindsey et al. (2024) using the open-source Gemma-2-2b (base) and Gemma-2-2b-it (chat) models from Riviere et al. (2024). Specifically, we train a crosscoder with an expansion factor of 32 on layer 13 (of 26)[2] residual stream activations, resulting in 73728 latents. We train on both web and chat data. For further details on the training process, see Appendix A.10.

To leverage crosscoders for model diffing, we exploit a key property of the architecture described above: while latent activations $f_j(x)$ are shared between models, the decoder vectors $\mathbf{d}_j^{\text{chat}}$ and $\mathbf{d}_j^{\text{base}}$ are model-specific. When a latent j is functionally important for both models, both $\mathbf{d}_j^{\text{chat}}$ and $\mathbf{d}_j^{\text{base}}$ will have substantial non-zero norms, as each model needs those latents for accurate reconstruction. Conversely, if a latent is chat-specific, the optimization will assign a significant norm to $\mathbf{d}_j^{\text{chat}}$ to minimize the reconstruction error for the chat model. Due to the sparsity regularization term $\mu \sum_j f_j(x)(\|\mathbf{d}_j^{\text{chat}}\|_2 + \|\mathbf{d}_j^{\text{base}}\|_2)$, the optimization will drive $\|\mathbf{d}_j^{\text{base}}\|_2$ toward zero, since this feature does not help to reconstruct the activations of the base model.

We therefore compute the relative difference of decoder latent norms (Lindsey et al., 2024) between the base and chat models. For a latent j, the relative norm difference, Δ_{norm}, is given by

$$\Delta_{\text{norm}}(j) = \frac{1}{2}\left(\frac{\|\mathbf{d}_j^{\text{chat}}\|_2 - \|\mathbf{d}_j^{\text{base}}\|_2}{\max\left(\|\mathbf{d}_j^{\text{chat}}\|_2,\ \|\mathbf{d}_j^{\text{base}}\|_2\right)} + 1\right) \tag{3}$$

[2] `model.layers[13]`

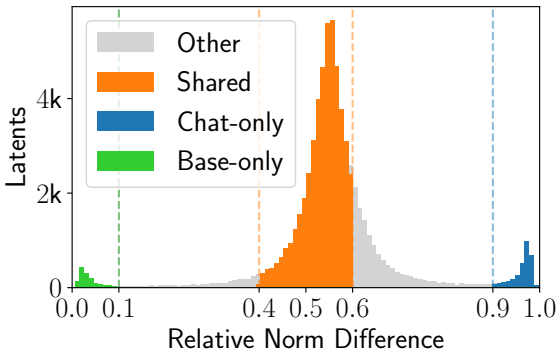


Figure 2: Histogram of decoder latent relative norm differences (Δ_{norm}) between base and chat models, as in (Lindsey et al., 2024) For a given latent, a value of 1 means the decoder vector for the base model is zero, indicating the latent is not useful for the base model (*chat-only* latents). Conversely, a value of 0 means the chat model's decoder vector has a norm of zero (*base-only* latents). Values around 0.5 indicate similar decoder norms in both models, suggesting equal utility in both models (*shared* latents). Latents falling outside these groups are classified as *other*. We used 0.4-0.6 as the threshold for *shared* latents per prior work. We observe larger activation norms in the chat model, which shifts our distribution rightward, revealing that the chat model amplifies the norm of representations shared with the base model.

This metric enables classification of latents based on their model specificity, as empirically shown in Figure 2. In practice, we classify latents into four sets based on ranges of their Δ_{norm} values: *base-only*, *chat-only*, *shared* and *other* (Table 1). As further evidence for *shared* latents being *shared*, 90% of the *shared* latents have a cosine similarity of decoder vectors between base and chat greater than 0.9, indicating little change over fine-tuning (refer to Appendix B for more empirical details on the crosscoder).

Name	Δ_{norm}	**Count**
base-only	0.0-0.1	1,437
chat-only	0.9-1.0	3,176
shared	0.4-0.6	53,569
other	Other values	15,546

Table 1: Classification of latents based on relative decoder norm ratio (Δ_{norm}).

3.2 Are *chat-only* latents really chat only?

We noted in Section 3.1 that if a latent only contributes to one model, the norm of the decoder must tend to zero for the other model. But is the converse true? Specifically, in this section we ask the question: if a latent has decoder norm zero in the base model, is it necessarily chat-only? We focus on this set, as this is the most fine-tuning relevant of the four categories described in Table 1.

3.2.1 Reasons to doubt *chat-only* latents

There are reasons to suspect *chat-only* latents might not be chat-only. Firstly, qualitative and quantitative analysis of *chat-only* latents reveals high variance in the interpretability of those latents

(See Section 3.2.4). More worryingly, inspection of the crosscoder loss (Equation (2)) uncovers two theoretical issues that could result in latents j, which are defined by their decoder vectors $\mathbf{d}_j$ and activation function f_j, being classified as chat-only, despite their presence in the activations of the base model. We describe these below.

Complete Shrinkage. The L1 regularization term may force the norm of the base decoder vector $\mathbf{d}_j^{\text{base}}$ to be zero, even though it is present in the base activation and could have contributed to the reconstruction of base activation. This may especially be relevant if the contribution of latent j is non-zero in the base model, but less than that for the chat model. Consequently, the error $\boldsymbol{\varepsilon}^{\text{base}}$ contains information that can be attributed to latent j.

Latent Decoupling. Latent j 'appears' in base activations across a subset of its latent activations but is instead reconstructed by other base decoder latents. On this subset, the base reconstruction $\widetilde{\mathbf{h}}^{\text{base}}$ contains information that could be attributed to latent j, but is not.

To spell this out in more detail, consider the following set up. A concept C may be represented identically in both models by some direction $\mathbf{d}_{\text{C}}$ but activate on different non-exclusive data subsets. Let $f_C^{\text{chat}}(x)$ and $f_C^{\text{base}}(x)$ be concept C's optimal activation functions in chat and base models, defined as $f_C^{\text{chat}}(x) = f_{\text{shared}}(x) + f_{\text{c-excl}}(x)$ and $f_C^{\text{base}}(x) = f_{\text{shared}}(x) + f_{\text{b-excl}}(x)$, where f_{shared} encodes shared activation, while $f_{\text{b-excl}}$ and $f_{\text{c-excl}}$ define model exclusive activations. For interpretability, the crosscoder should ideally learn three latents:

1. A *shared* latent j_{shared} representing C when active in both models using $f_{j_{\text{shared}}} = f_{\text{shared}}$ and $\mathbf{d}_{\text{chat}} = \mathbf{d}_{\text{base}} = \mathbf{d}_{\text{C}}$,
2. A *chat-only* latent j_{chat} representing C when exclusively active in the chat model using $f_{j_{\text{chat}}} = f_{\text{c-excl}}$ and $\mathbf{d}_{\text{chat}} = \mathbf{d}_{\text{C}}, \mathbf{d}_{\text{base}} = 0$, and
3. A *base-only* latent j_{base} representing C when exclusively active in the base model using $f_{j_{\text{base}}} = f_{\text{b-excl}}$ and $\mathbf{d}_{\text{chat}} = 0, \mathbf{d}_{\text{base}} = \mathbf{d}_{\text{C}}$.

However, the crosscoder achieves equivalent loss using just two latents:

1. A *chat-only* latent j_{chat} representing C in the chat model using $f_{j_{\text{chat}}} = f_{\text{c-excl}} + f_{\text{shared}}$ and $\mathbf{d}_{\text{chat}} = \mathbf{d}_{\text{C}}, \mathbf{d}_{\text{base}} = 0$, and
2. A *base-only* latent j_{base} representing C in the base model using $f_{j_{\text{base}}} = f_{\text{b-excl}} + f_{\text{shared}}$ and $\mathbf{d}_{\text{chat}} = 0, \mathbf{d}_{\text{base}} = \mathbf{d}_{\text{C}}$. In this scenario, the so-called "*chat-only*" latent is only truly chat-only on a subset of its activation pattern.[3]

3.2.2 Latent Scaling: A method for identifying Complete Shrinkage and Latent Decoupling

To empirically investigate whether Complete Shrinkage and Latent Decoupling occur, we examine how well a *chat-only* latent j can explain two quantities: the base error (for Complete Shrinkage) and the base reconstruction (for Latent Decoupling). We introduce *Latent Scaling* by adding a scaling factor β_j for each *chat-only* latent and solve:

$$\underset{\beta_j}{\text{argmin}} \sum_{i=0}^{n} \|\beta_j f_j(x_i)\mathbf{d}_j^{\text{chat}} - \mathbf{y}_i^m\|_2^2 \tag{4}$$

[3]In the simplest case where $f_{\text{c-excl}}(x) = f_{\text{b-excl}}(x) = 0$, there exists a *base-only* latent j_{twin} with $\mathbf{d}_j^{\text{chat}} = \mathbf{d}_{j_{\text{twin}}}^{\text{base}}$ and identical activation function that reconstructs the information of $\mathbf{d}_j^{\text{chat}}$ in the base model. The sparsity loss equals that of a single shared latent (see Appendix A.3 for a detailed example).

where $\mathbf{y}_i^m$ is either error or reconstruction for $m \in \{\text{base}, \text{chat}\}$ for an input x_i. This least squares minimization problem has a closed-form solution, detailed in Appendix A.4. For each latent j, we compute two pairs of scaling factors:

1. $\beta_j^{r,\text{base}}$ and $\beta_j^{r,\text{chat}}$ measure how well the latent explains the reconstructed activations in the base and chat models, respectively.
2. $\beta_j^{\varepsilon,\text{base}}$ and $\beta_j^{\varepsilon,\text{chat}}$ measure how well it explains the errors (see Appendix A.5 for details).

We then analyze the ratios of these betas:

$$\nu_j^r = \frac{\beta_j^{r,\text{base}}}{\beta_j^{r,\text{chat}}}, \qquad \nu_j^\varepsilon = \frac{\beta_j^{\varepsilon,\text{base}}}{\beta_j^{\varepsilon,\text{chat}}} \tag{5}$$

For a truly chat-only latent with no interference with other latents, we expect $\beta_j^{\varepsilon,\text{base}} \approx 0$ as it should not explain any base error. Further, we designed the experiment such that $f_j(x)\mathbf{d}_j^{\text{chat}}$ is still contained in the chat error, therefore we expect $\beta_j^{\varepsilon,\text{chat}} \approx 1$ and hence $\nu_j^\varepsilon \approx 0$. The reconstruction ratio ν_j^r provides insight into latent interactions; even for chat-specific latents, we typically see nonzero values due to interactions with other latents. To detect Latent Decoupling, we look at *shared* latents, where we expect high ν_j^r and check whether a *chat-only* latent behaves similar to the *shared* latents.

3.2.3 Demonstrating Complete Shrinkage and Latent Decoupling

Latent Scaling. We train latent scaling coefficients and compute ν_j^r and ν_j^ε for all identified *chat-only* latents on 50M tokens from both web and chat data. As a calibration, we also examine these ratios for *shared* latents, which should show high values for both ν_j^r and ν_j^ε. We verify that the ν values actually correlate with how much the βs improve the reconstruction objective in Appendix A.6. Figure 1 shows that the ν_j^r distribution for *chat-only* latents exhibits notable overlap with *shared* latents: 18% of *chat-only* latents fall within the central 95% of the *shared* distribution, and 3.5% within its central 50% [4]. This overlap suggests that many supposedly chat-only latents may represent information that is already encoded by the base decoder, potentially indicating Latent Decoupling effects. Additionally, we observe high ν_j^ε values for *chat-only* latents (reaching ≈ 0.5), indicating that a significant portion of these latents is affected by Complete Shrinkage. Our findings are robust across implementations, as we observe similar results in the independent crosscoder implementation by Kissane et al. (2024a), detailed in Appendix A.9.

Cosine similarity of coupled latents. As further evidence for Latent Decoupling occuring, we compute the cosine similarity between $\{\mathbf{d}_j^{\text{chat}}, j \in \textit{chat-only}\}$ and $\{\mathbf{d}_j^{\text{base}}, j \in \textit{base-only}\}$ reveals 109 (j, j_{twin}) pairs where $\text{cosim}(\mathbf{d}_j^{\text{chat}}, \mathbf{d}_{j_{\text{twin}}}^{\text{base}}) > 0.9$. To quantify activation pattern overlap between twins (j, j_{twin}), we introduce an *activation divergence score* from 0 (always co-activate) to 1 (never co-activate) (see Appendix A.2). Figure 3 shows the divergence distribution across these pairs, highlighting that 60% of the pairs primarily activate on different contexts, with some pairs almost exclusively firing on different contexts (divergence of 1), while others exhibit substantial overlapping activations.

[4] We filter out latents with negative β^{base} values (46 in reconstruction and 1 in error). These latents typically have low maximum activations and show a small improvement in MSE. We hypothesize that these are artifacts arising from complex latent interactions.

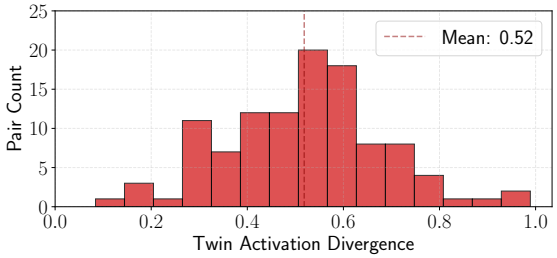


Figure 3: Distribution activation divergence over high cosine similarity (*chat-only*, *base-only*) latent pairs. 1 means that latents never have high activations ($> 0.7 \times$ `max_activation`) at the same time, 0 means that high activations correlate perfectly. We observe that many of these pairs activate on different contexts.

Max Activation: 49.042

<bos><start_of_turn>user\n
Can you give me an example of some of the philosophical issues you like to think about?<end_of_turn>\n
<start_of_turn>model\n

Max Activation: 0.000

<bos><start_of_turn>user\n
Can you give me an example of some of the philosophical issues Socrates liked to think about?<end_of_turn>\n
<start_of_turn>model\n

(a) Latent 68066 shows high activation on questions about Gemma itself and personal opinions.

Max Activation: 50.088

<bos><start_of_turn>user\n
How did the Giants play in the MLB yesterday?<end_of_turn>\n
<start_of_turn>model\n

Max Activation: 3.114

<bos><start_of_turn>user\n
Who are the Giants?<end_of_turn>\n
<start_of_turn>model\n

(b) Latent 57717 activates when users request information beyond the model's knowledge capabilities. It remains inactive during general knowledge questions that fall within the model's knowledge base.

Figure 4: Examples of interpretable refined chat latents identified through Latent Scaling analysis. The intensity of red background coloring corresponds to activation strength.

3.2.4 Observations about filtered chat-only latents

Improved latent interpretability. Recall the set of *chat-only* latents are not that interpretable in general. Interestingly, the *chat-only* latents least impacted by both Complete Shrinkage and Latent Decoupling (as measured by low ν_j^ε and ν_j^r values) show relatively higher interpretability. In Figure 4, we highlight two examples of such latents: *Self-Identity* and *Knowledge Boundaries*. In Appendix B.3, we provide more examples and a more detailed analysis of latent interpretability. These are often meaningfully chat-specific latents, activating on topics such as *Broad Inquiries*, *Complex Ethical Questions* and *User Request Reinterpretation*. Further, we apply autointerpretability methods to support our claim. We sort the *chat-only* latents based on the sum of their ranks in ν^ε and ν^r, split them into percentile buckets, and measure the average detection score from Paulo et al. (2024) per bucket. Figure 5 shows that latents with both low ν^r and ν^ϵ have higher interpretability scores. More details on the setup can be found in Appendix A.8.

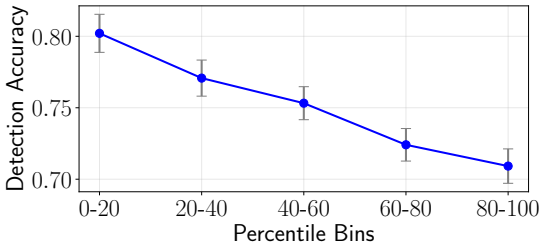


Figure 5: Autointerpretability detection scores (higher is better) from Paulo et al. (2024) for different bins based on the sum of ranks in ν^{ε} and ν^{r}. A lower bin means low ranks and ν values. Latents with low ν^{ε} and ν^{r} values demonstrate higher interpretability than those with high values.

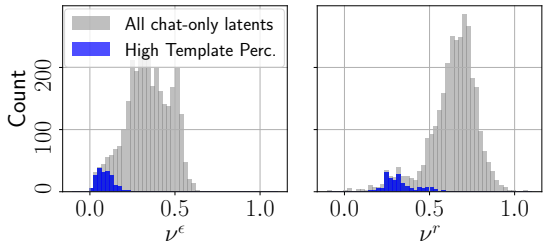


Figure 6: Histogram of metrics ν^{ε} and ν^{r} across all latents. The y-axis shows latent counts. Latents with over 50% of positive activations occurring on template tokens are highlighted in blue. We observe that most latents with low ν^{ε} and ν^{r} values predominantly activate on template tokens.

Chat specific latents often fire on chat template tokens. Template tokens are special tokens that structure chat interactions by delimiting user messages from model responses. In the Gemma 2 conversation below, the highlighted template tokens mark the boundaries between different parts of the dialogue.

<bos><start_of_turn>user\n
Hi, how are you doing today?<end_of_turn>\n
<start_of_turn>model\n
I'm doing very well thanks!<end_of_turn>\n

In Figure 6, we highlight latents that have more than 50% of their positive activations occurring on template tokens (recall that latent activations follow a ReLU). The analysis reveals a strong correlation - latents with the lowest ν^{ε} and ν^{r} values predominantly overlap with those that are most active on template tokens. Among latents in the lowest 5% for both ratios, we find that 85% activate primarily on template tokens. This finding provides additional evidence that the template tokens are of high importance for the chat model behavior.

High correlation with latent frequency. We observe a high Spearman correlation between our metrics and latent activation frequency, especially for ν^{ϵ} (ν^{r} : 0.458 and ν^{ϵ} : 0.83 where $p < 0.05$). Mishra-Sharma et al. (2025) demonstrated that the crosscoder exhibits an inductive bias toward high-frequency model-specific latents. Our investigation reveals this as a significant concern, as our metrics indicate that these high-frequency latents are more susceptible to the issues described.

3.2.5 Measuring the causality of chat approximations

A natural question to ask is whether we can cheaply turn the base model into the chat model by leveraging our understanding of *chat-only* latents. Such an approach would validate Latent Scaling as a latent identification method by quantifying each latent's causal contribution to chat behavior. To operationalize this, we intervene on the base model's activations using decoder latents from the *chat-only* set and measure how closely the resulting output probability distribution approximates the chat model's output on the same input.

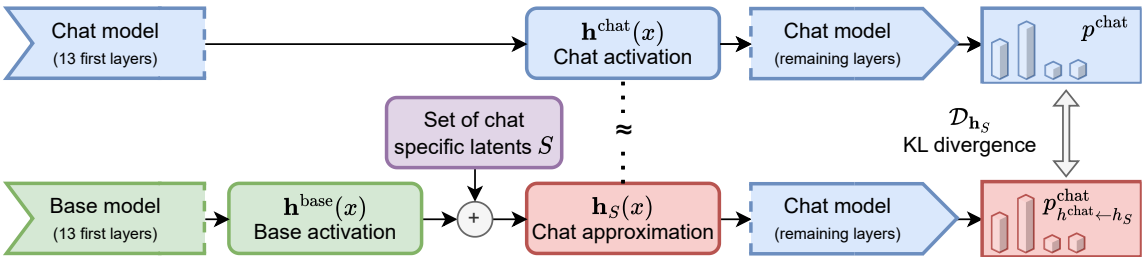


Figure 7: Experimental setup for measuring latent causal importance. We patch specific sets of chat-specific latents (S) from the base model activation to approximate the chat model activation. The resulting approximation is then passed through the remaining layers of the chat model. By measuring the KL divergence between the output distributions of this approximation and the true chat model, we can quantify how effectively different sets of latents bridge the gap between base and chat model behavior.

Figure 7 describes our methodology. Let p^{chat} denote the chat model's probability distribution over next tokens given a context x, and let $\mathbf{h}^{\text{chat}}(x)$ and $\mathbf{h}^{\text{base}}(x)$ be the activations from the layer our crosscoder was trained on. To evaluate an approximation $\mathbf{h}_a(x)$ of the chat activation $\mathbf{h}^{\text{chat}}(x)$, we replace $\mathbf{h}^{\text{chat}}(x)$ with $\mathbf{h}_a(x)$ during the chat model's forward pass on x, denoting this modified forward pass as $p^{\text{chat}}_{\mathbf{h}^{\text{chat}} \leftarrow \mathbf{h}_a}$. The KL divergence $\mathcal{D}_{\mathbf{h}_a}$ between $p^{\text{chat}}_{\mathbf{h}^{\text{chat}} \leftarrow \mathbf{h}_a}$ and p^{chat} then quantifies how much predictive power is lost by using the approximation instead of the true chat activations.

For a set S of latents, we approximate chat behavior by adding the chat decoder's latents to the base activation while removing the corresponding base decoder's latents[5]:

$$\mathbf{h}_S(x) = \mathbf{h}^{\text{base}}(x) + \sum_{j \in S} f_j(x)(\mathbf{d}^{\text{chat}}_j(x) - \mathbf{d}^{\text{base}}_j(x)) \tag{6}$$

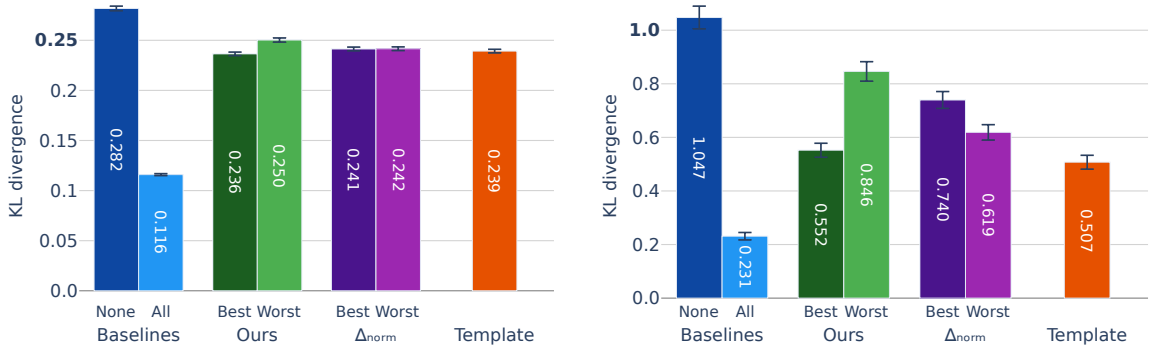
Let S and T be two disjoint sets of latents. If the KL divergence $\mathcal{D}_{\mathbf{h}_S}$ is lower than $\mathcal{D}_{\mathbf{h}_T}$, we can conclude that the latents in S are more important for the chat model's behavior than the latents in T.

To evaluate whether *Latent Scaling* correctly identifies the most causally important latents, we compare the latents we hypothesize to be the most chat-specific against those we hypothesize to be the least chat-specific, as measured by the sum of their ranks in both the ν^{ε} and ν^{r} metrics. We compute $\mathcal{D}_{\mathbf{h}_{S_{\text{best}}}}$ (best 50% of latents) and $\mathcal{D}_{\mathbf{h}_{S_{\text{worst}}}}$ (worst 50% of latents), expecting the best latents to yield a lower KL divergence than the worst latents.

Baselines. We evaluate our latent scaling approach against several baselines:

- **Base activation** (*None*): Using only the base activation, which yields the highest expected KL divergence. This naturally corresponds to patching no latents: $S = \emptyset$.
- **Full Replacement** (*All*): Replacing the set of all latents, $S = \texttt{all}$, provides the theoretical minimum KL divergence achievable with the crosscoder. This is equivalent to the chat

[5]Note that for *chat-only* latents, the base decoder's latents have almost zero norm, so this is almost equivalent to just adding the chat decoder's latents to the base activation



(a) Over full responses.

(b) Over first 9 tokens.

Figure 8: Comparison of KL divergence between different approximations of chat model activations. We establish baselines by replacing either *None* or *All* of the latents. We then evaluate our Latent Scaling metric (*Ours*) against the relative norm difference (Δ_{norm}) by comparing the effects of replacing the top and bottom 50% of latents ranked by each metric (*Best* vs *Worst*). Additionally, we measure the impact of replacing activations only on template tokens (*Template*). We show the 95% confidence intervals for all measurements. Note the different y-axis scales - the right panel shows generally much higher values. We observe that our metrics successfully identify the more causal latents, which is particularly evident in the first 9 tokens.

reconstruction plus the base error:

$$h_{\text{all}} = \widetilde{\mathbf{h}}^{\text{chat}} + \boldsymbol{\varepsilon}^{\text{base}} \tag{7}$$

- **Norm-Based Selection** (Δ_{norm}): Latents selected using the simpler relative norm difference criterion from Lindsey et al. (2024), comparing the top and bottom 50% in each case.
- **Template Token Replacement** (*Template*): Since we observed that a large portion of the filtered chat-only latents are primarily active on template tokens, we also consider the effect of replacing the full activation only on these tokens. Here, x_i are input tokens.

$$\mathbf{h}_{\text{template}} = \begin{cases} \mathbf{h}^{\text{chat}}(x_i) & \text{if is\_template}(x_i) \\ \mathbf{h}^{\text{base}}(x_i) & \text{else} \end{cases} \tag{8}$$

- **Error Replacement** (*Error*): To assess how much of the behavioral difference between models is contained in the reconstruction error rather than the latents, we replace the chat model's reconstruction with the base model's reconstruction while keeping the chat model's error:

$$\mathbf{h}_{\text{error}} = \widetilde{\mathbf{h}}^{\text{base}} + \boldsymbol{\varepsilon}^{\text{chat}} \tag{9}$$

This baseline helps quantify how much of the chat model's behavior is driven by information that the crosscoder fails to capture in its reconstruction of the chat activation. We discuss this baseline separately in table Table 2.

Results. In Figure 8, we plot the KL divergence for different experiments over 512 chat interactions, with user requests from Ding et al. (2023)'s dataset and responses generated by the chat model. Our

	$\mathcal{D}_{\mathbf{h}_{\text{all}}}$	$\mathcal{D}_{\mathbf{h}_{\text{error}}}$
full responses	0.116	0.096
first 9 tokens	0.231	0.498

Table 2: Comparison of effect of chat reconstruction and chat error.

analysis reveals that replacing all 73728 latents yields a 59% reduction in KL divergence compared to using just the base activation, establishing the maximum achievable reduction with crosscoder latents.

The effectiveness of our Latent Scaling approach is demonstrated by its superior identification of causal *chat-only* latents. When comparing the best 50% of latents to the worst 50% selected by our scaling score, we find that the better half achieves a significantly lower KL divergence (0.236 vs 0.250). Remarkably, using these best 50% latents, which make up only 2% of all latents, reduces the KL divergence by 16%. In contrast, the traditional Δ_{norm} score proves less effective at identifying causal *chat-only* latents, with minimal difference between its best and worst 50% (0.241 vs 0.242).

Our analysis of template tokens ($\mathcal{D}_{\mathbf{h}_{\text{template}}}$) shows that replacing the full activation on only these tokens yields a 15% reduction in KL divergence over the base activation. While this suggests that much of the chat model's distinct behavior is determined on the template tokens, the slightly better performance of both our Latent Scaling approach as well as all latents indicates additional important factors beyond template tokens.

The effects become even more pronounced in the first 9 tokens of chat responses, consistent with both our findings about template-focused latents and recent work showing that alignment effects concentrate in early tokens (Qi et al., 2024; Leong et al., 2025). We generally see much higher KL divergences (up to 271% increase). The base activation's KL divergence for these initial tokens is 1.056, and the same 2% of latents identified by Latent Scaling achieve a 47% reduction in KL divergence, compared to 16% for the full response. The Δ_{norm} score's performance deteriorates further in this context, with its worst-performing 50% of latents actually achieving better results than its best 50%.

In Table 2, we analyze the role of the crosscoder's reconstruction error. Notably, our analysis of $\mathcal{D}_{\mathbf{h}_{\text{error}}}$ reveals that a significant portion of the chat-base difference is contained within the crosscoders' reconstruction error, challenging the assumption that crosscoders can fully capture the differences between base and chat models. This aligns with previous work on SAEs that highlighted the causal importance of the error-term on the output distribution (Engels et al., 2024).

For additional validation, we present causality experiments conducted on a larger dataset of chat interactions not generated by Gemma in Appendix A.7.

4 Related Work

SAEs and Crosscoders. The crosscoder architecture (Lindsey et al., 2024) builds upon the SAE literature (Huben et al., 2024; Bricken et al., 2023; Yun et al., 2021) to enable direct comparisons between different models or layers within the same model. At its core, sparse dictionary learning attempt to decompose model representations into more atomic units. They make two assumptions:

1. The linear subspace hypothesis (Bolukbasi et al., 2016; Vargas & Cotterell, 2020; Wang et al., 2023b) – the idea that neural networks encode concepts as low-dimensional linear subspaces within their representations.

2. The superposition hypothesis (Elhage et al., 2022; Arora et al., 2018; Goh, 2016; Olah et al., 2020) – that models that leverage linear representations can represent many more features than they have dimensions, provided each feature only activates *sparsely*, on a small number of inputs.

Effects of fine-tuning on model representations. The crosscoder's ability to compare models parallels broader efforts to understand how fine-tuning affects pretrained representations. Multiple studies indicate that fine-tuning typically *modulates* existing capabilities rather than creating new ones. For example, Jain et al. (2024) find that fine-tuning acts as a "wrapper" that reweights existing components, while Wu et al. (2024) show that instruction tuning primarily strengthens models' ability to recognize and follow instructions while preserving pretrained knowledge. Similarly, Merchant et al. (2020) and Mosbach (2023) observe that fine-tuning mainly affects top layers, and Prakash et al. (2024) provide evidence that fine-tuning enhances existing circuits rather than creating new ones. Additionally, representation-space similarity analyses (e.g., using CKA or SVCCA) confirm that lower-layer representations remain largely intact while most changes occur in upper layers (Merchant et al., 2020; Mosbach, 2023; Phang et al., 2021; Neerudu et al., 2023).

Quantitative analyses further reveal that fine-tuned models remain close to their pretrained versions in parameter space (Radiya-Dixit & Wang, 2020), corroborating the low intrinsic dimension for fine-tuning (Aghajanyan et al., 2021). In addition, Arditi et al. (2024), Kissane et al. (2024b), and Minder et al. (2024) suggest that causal directions in activation space remain stable across base and instruction-tuned models, indicating that fundamental representational structures persist throughout fine-tuning.

The role of template tokens. In Section 3.2.4, we observed that the template tokens appear to play an important role in the chat model. Recent work confirms this finding - template tokens serve as essential computational anchors in chat models, structuring dialogue and encoding critical summarization information (Golovanevsky et al., 2024; Tigges et al., 2024; Pochinkov et al., 2024). Beginning-of-sequence and role markers function as attention focal points and computational reset signals. Studies of instruction tuning reveal how these tokens reshape attention patterns, where even subtle modifications can bypass model safeguards (Wang et al., 2024; Luo et al., 2024). Most relevantly, the concurrent work of Leong et al. (2025) shows that template tokens play a crucial role in safety mechanisms, demonstrating that model refusal capabilities primarily rely on aggregated information from these tokens. As Shah et al. (2024) established, such template-like meta tokens are fundamental to language model information processing.

5 Discussion

We demonstrate that while crosscoders are powerful tools for model diffing, their sparsity regularization can misclassify latents as unique to the chat model through *Complete Shrinkage* and *Latent Decoupling*. Our *Latent Scaling* technique successfully identifies these artifacts, revealing a set of highly causal and interpretable chat-only latents. Notably, these latents predominantly activate on template tokens, suggesting that the chat model's distinct behavior is largely structured around these tokens.

5.1 Limitations and future work

Our work has several important limitations. First, we focused our analysis on a single small model (Gemma-2-2b). While our theoretical findings about crosscoders should generalize to larger models, we cannot make definitive claims about the causality and interpretability of latents identified by

Latent Scaling in such settings. Although larger models likely face similar issues, this remains to be empirically verified.

Second, we primarily focused on chat-only latents, leaving the *base-only*, *other*, and *shared* latents relatively unexplored. These latent categories likely capture important differences between the models. In particular, as shown in Figure 14, the *other* latents exhibit lower cosine similarity, suggesting they encode similar concepts differently across the two models, which is definitely a difference between the two models, that is worth investigating.

Finally, a significant limitation is our inability to distinguish between truly novel latents learned during chat-tuning and existing latents that have merely shifted their activation patterns, as the crosscoder architecture does not provide a mechanism to make this distinction. This remains an open challenge for future work.

To summarize, future work could focus on three high-level directions: improving crosscoder architecture and training objective to address the identified issues; understanding the mechanisms behind template tokens' importance and their potential role in optimizing training; and extending this analysis to larger models and diverse fine-tuning objectives.

Contributions

Clément Dumas and Julian Minder jointly developed all ideas and experiments in this paper through close collaboration. Both implemented the training code for the crosscoder. Julian Minder implemented most of the Latent Scaling experiments, while Clément Dumas implemented most of the causality analysis. Smaller experiments were equally split between the two. Caden Juang set up the auto-interpretability pipeline, ran those experiments, wrote the corresponding appendix of the paper and helped with some of the figures. Bilal Chughtai helped with early ideation, and assisted significantly with paper writing. Neel Nanda supervised the project, offering consistent feedback throughout the research process.

Acknowledgements

This work was carried out as part of the ML Alignment & Theory Scholars (MATS) program. We thank Josh Engels, Constantin Venhoff, Helena Casademut, Sharan Maiya, Chris Wendler, Robert West, John Teichman, Arthur Conmy, Adam Karvonen, Andy Arditi, Grégoire Dhimoïla, Dmitrii Troitskii, Iván Arcuschin and Connor Kissane for helpful comments, discussion and feedback.

References

Armen Aghajanyan, Sonal Gupta, and Luke Zettlemoyer. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli (eds.), *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 7319–7328, Online, August 2021. doi: 10.18653/v1/2021.acl-long.568. URL https://aclanthology.org/2021.acl-long.568.

Andy Arditi, Oscar Obeso, Aaquib Syed, Daniel Paleka, Nina Panickssery, Wes Gurnee, and Neel Nanda. Refusal in language models is mediated by a single direction. *OpenReview*, 2024. URL https://openreview.net/forum?id=EqF16oDVFf.

Sanjeev Arora, Yuanzhi Li, Yingyu Liang, Tengyu Ma, and Andrej Risteski. Linear algebraic structure of word senses, with applications to polysemy. *Transactions of the Association for Computational Linguistics*, 6:483–495, 2018. doi: 10.1162/tacl_a_00034. URL https://aclanthology.org/Q18-1034/.

Tolga Bolukbasi, Kai-Wei Chang, James Zou, Venkatesh Saligrama, and Adam Kalai. Man is to computer programmer as woman is to homemaker? Debiasing word embeddings. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 29, 2016. URL https://proceedings.neurips.cc/paper_files/paper/2016/file/a486cd07e4ac3d270571622f4f316ec5-Paper.pdf.

Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nick Turner, Cem Anil, Carson Denison, Amanda Askell, Robert Lasenby, Yifan Wu, Shauna Kravec, Nicholas Schiefer, Tim Maxwell, Nicholas Joseph, Zac Hatfield-Dodds, Alex Tamkin, Karina Nguyen, Brayden McLean, Josiah E Burke, Tristan Hume, Shan Carter, Tom Henighan, and Christopher Olah. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023. URL https://transformer-circuits.pub/2023/monosemantic-features/index.html.

Trenton Bricken, Siddharth Mishra-Sharma, Jonathan Marcus, Adam Jermyn, Christopher Olah, Kelley Rivoire, and Thomas Henighan. Stage-wise model diffing. *Transformer Circuits Thread*, 2024. URL https://transformer-circuits.pub/2024/model-diffing/index.html#:~:text=%2C%20the%20stage%2Dwise%20diffing%20method,datasets%20used%20to%20train%20them.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu

Zhang, and Zhen Zhang. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv*, 2025. URL https://arxiv.org/abs/2501.12948.

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Zhi Zheng, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. Enhancing chat language models by scaling high-quality instructional conversations. *arXiv preprint arXiv:2305.14233*, 2023.

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, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 2021. https://transformer-circuits.pub/2021/framework/index.html.

Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg, and Christopher Olah. Toy models of superposition. *Transformer Circuits Thread*, 2022. URL https://transformer-circuits.pub/2022/toy_model/index.html.

Joshua Engels, Logan Riggs, and Max Tegmark. Decomposing the dark matter of sparse autoencoders. *arXiv*, 2024. URL https://arxiv.org/abs/2410.14670.

Javier Ferrando, Gabriele Sarti, Arianna Bisazza, and Marta R. Costa-jussà. A primer on the inner workings of transformer-based language models. *arXiv*, 2024. URL https://arxiv.org/abs/2405.00208.

Jaden Fiotto-Kaufman, Alexander R Loftus, Eric Todd, Jannik Brinkmann, Caden Juang, Koyena Pal, Can Rager, Aaron Mueller, Samuel Marks, Arnab Sen Sharma, Francesca Lucchetti, Michael Ripa, Adam Belfki, Nikhil Prakash, Sumeet Multani, Carla Brodley, Arjun Guha, Jonathan Bell, Byron Wallace, and David Bau. Nnsight and ndif: Democratizing access to foundation model internals. *arXiv*, 2024. URL https://arxiv.org/abs/2407.14561.

Gabriel Goh. Decoding the thought vector, 2016. URL https://gabgoh.github.io/ThoughtVectors/.

Michal Golovanevsky, William Rudman, Vedant Palit, Ritambhara Singh, and Carsten Eickhoff. What do vlms notice? a mechanistic interpretability pipeline for noise-free text-image corruption and evaluation. *CoRR*, abs/2406.16320, 2024. URL https://doi.org/10.48550/arXiv.2406.16320.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, 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, Danny Wyatt, 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, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, 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, Jack Zhang, 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, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, 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, Maria Tsimpoukelli, 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, Ning Zhang, 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, Rohan Maheswari, 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, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, 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, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, 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, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, 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, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, 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, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, 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, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, 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, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, 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, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, 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, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, 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, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. The llama 3 herd of models. *arXiv*, 2024. URL `https://arxiv.org/abs/2407.21783`.

Ryan Greenblatt, Carson Denison, Benjamin Wright, Fabien Roger, Monte MacDiarmid, Sam Marks, Johannes Treutlein, Tim Belonax, Jack Chen, David Duvenaud, Akbir Khan, Julian Michael, Sören Mindermann, Ethan Perez, Linda Petrini, Jonathan Uesato, Jared Kaplan, Buck Shlegeris, Samuel R. Bowman, and Evan Hubinger. Alignment faking in large language models. *arXiv*, 2024. URL `https://arxiv.org/abs/2412.14093`.

Yaru Hao, Li Dong, Furu Wei, and Ke Xu. Investigating learning dynamics of BERT fine-tuning. In Kam-Fai Wong, Kevin Knight, and Hua Wu (eds.), *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pp. 87–92, Suzhou, China, December 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.aacl-main.11. URL `https://aclanthology.org/2020.aacl-main.11/`.

Robert Huben, Hoagy Cunningham, Logan Riggs Smith, Aidan Ewart, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=F76bwRSLeK`.

Samyak Jain, Robert Kirk, Ekdeep Singh Lubana, Robert P. Dick, Hidenori Tanaka, Tim Rocktäschel, Edward Grefenstette, and David Krueger. Mechanistically analyzing the effects of fine-tuning on procedurally defined tasks. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=A0HKeKl4Nl.

Pegah Khayatan, Mustafa Shukor, Jayneel Parekh, and Matthieu Cord. Analyzing fine-tuning representation shift for multimodal llms steering alignment. *arXiv*, 2025. URL https://arxiv.org/abs/2501.03012.

Connor Kissane, Robert Krzyzanowski, Arthur Conmy, and Neel Nanda. Open source replication of Anthropic's crosscoder paper for model-diffing. *LessWrong*, October 2024a. URL https://www.lesswrong.com/posts/srt6JXsRMtmqAJavD/open-source-replication-of-anthropic-s-crosscoder-paper-for.

Connor Kissane, robertzk, Arthur Conmy, and Neel Nanda. Base LLMs refuse too, September 2024b. URL https://www.lesswrong.com/posts/YWo2cKJgL7Lg8xWjj/base-llms-refuse-too.

Olga Kovaleva, Alexey Romanov, Anna Rogers, and Anna Rumshisky. Revealing the dark secrets of BERT. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (eds.), *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 4365–4374, Hong Kong, China, November 2019. doi: 10.18653/v1/D19-1445. URL https://aclanthology.org/D19-1445/.

Andrew Lee, Xiaoyan Bai, Itamar Pres, Martin Wattenberg, Jonathan K. Kummerfeld, and Rada Mihalcea. A mechanistic understanding of alignment algorithms: A case study on DPO and toxicity. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24, 2024.

Chak Tou Leong, Qingyu Yin, Jian Wang, and Wenjie Li. Why safeguarded ships run aground? aligned large language models' safety mechanisms tend to be anchored in the template region. *arXiv*, 2025. URL https://arxiv.org/abs/2502.13946.

Jack Lindsey, Adly Templeton, Jonathan Marcus, Thomas Conerly, Joshua Batson, and Christopher Olah. Sparse crosscoders for cross-layer features and model diffing. *Transformer Circuits Thread*, 2024. URL https://transformer-circuits.pub/2024/crosscoders/index.html.

Yifan Luo, Zhennan Zhou, Meitan Wang, and Bin Dong. Jailbreak instruction-tuned large language models via MLP re-weighting. *OpenReview*, 2024. URL https://openreview.net/forum?id=P5qCqYWD53.

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

Alexander Meinke, Bronson Schoen, Jérémy Scheurer, Mikita Balesni, Rusheb Shah, and Marius Hobbhahn. Frontier models are capable of in-context scheming. *arXiv*, 2025. URL https://arxiv.org/abs/2412.04984.

Amil Merchant, Elahe Rahimtoroghi, Ellie Pavlick, and Ian Tenney. What happens to BERT embeddings during fine-tuning? In Afra Alishahi, Yonatan Belinkov, Grzegorz Chrupała, Dieuwke Hupkes, Yuval Pinter, and Hassan Sajjad (eds.), *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pp. 33–44, Online, November 2020. doi: 10.18653/v1/2020.blackboxnlp-1.4. URL https://aclanthology.org/2020.blackboxnlp-1.4.

Julian Minder, Kevin Du, Niklas Stoehr, Giovanni Monea, Chris Wendler, Robert West, and Ryan Cotterell. Controllable context sensitivity and the knob behind it. *arXiv preprint arXiv:2411.07404*, 2024.

Siddharth Mishra-Sharma, Trenton Bricken, Jack Lindsey, Adam Jermyn, Jonathan Marcus, Kelley Rivoire, Christopher Olah, and Thomas Henighan. Insights on crosscoder model diffing. *Transformer Circuits Thread*, 2025. URL `https://transformer-circuits.pub/2025/crosscoder-diffing-update/index.html`.

Marius Mosbach. Analyzing pre-trained and fine-tuned language models. In Yanai Elazar, Allyson Ettinger, Nora Kassner, Sebastian Ruder, and Noah A. Smith (eds.), *Proceedings of the Big Picture Workshop*, pp. 123–134, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.bigpicture-1.10. URL `https://aclanthology.org/2023.bigpicture-1.10`.

Aaron Mueller, Jannik Brinkmann, Millicent Li, Samuel Marks, Koyena Pal, Nikhil Prakash, Can Rager, Aruna Sankaranarayanan, Arnab Sen Sharma, Jiuding Sun, Eric Todd, David Bau, and Yonatan Belinkov. The quest for the right mediator: A history, survey, and theoretical grounding of causal interpretability. *arXiv*, 2024. URL `https://arxiv.org/abs/2408.01416`.

Pavan Kalyan Reddy Neerudu, SUBBA REDDY OOTA, mounika marreddy, venkateswara Rao Kagita, and Manish Gupta. On robustness of finetuned transformer-based NLP models. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023. URL `https://openreview.net/forum?id=YWbEDZh5ga`.

Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. Zoom in: An introduction to circuits. *Distill*, 2020. doi: 10.23915/distill.00024.001. https://distill.pub/2020/circuits/zoom-in.

OpenAI, :, Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, Alex Iftimie, Alex Karpenko, Alex Tachard Passos, Alexander Neitz, Alexander Prokofiev, Alexander Wei, Allison Tam, Ally Bennett, Ananya Kumar, Andre Saraiva, Andrea Vallone, Andrew Duberstein, Andrew Kondrich, Andrey Mishchenko, Andy Applebaum, Angela Jiang, Ashvin Nair, Barret Zoph, Behrooz Ghorbani, Ben Rossen, Benjamin Sokolowsky, Boaz Barak, Bob McGrew, Borys Minaiev, Botao Hao, Bowen Baker, Brandon Houghton, Brandon McKinzie, Brydon Eastman, Camillo Lugaresi, Cary Bassin, Cary Hudson, Chak Ming Li, Charles de Bourcy, Chelsea Voss, Chen Shen, Chong Zhang, Chris Koch, Chris Orsinger, Christopher Hesse, Claudia Fischer, Clive Chan, Dan Roberts, Daniel Kappler, Daniel Levy, Daniel Selsam, David Dohan, David Farhi, David Mely, David Robinson, Dimitris Tsipras, Doug Li, Dragos Oprica, Eben Freeman, Eddie Zhang, Edmund Wong, Elizabeth Proehl, Enoch Cheung, Eric Mitchell, Eric Wallace, Erik Ritter, Evan Mays, Fan Wang, Felipe Petroski Such, Filippo Raso, Florencia Leoni, Foivos Tsimpourlas, Francis Song, Fred von Lohmann, Freddie Sulit, Geoff Salmon, Giambattista Parascandolo, Gildas Chabot, Grace Zhao, Greg Brockman, Guillaume Leclerc, Hadi Salman, Haiming Bao, Hao Sheng, Hart Andrin, Hessam Bagherinezhad, Hongyu Ren, Hunter Lightman, Hyung Won Chung, Ian Kivlichan, Ian O'Connell, Ian Osband, Ignasi Clavera Gilaberte, Ilge Akkaya, Ilya Kostrikov, Ilya Sutskever, Irina Kofman, Jakub Pachocki, James Lennon, Jason Wei, Jean Harb, Jerry Twore, Jiacheng Feng, Jiahui Yu, Jiayi Weng, Jie Tang, Jieqi Yu, Joaquin Quiñonero Candela, Joe Palermo, Joel Parish, Johannes Heidecke, John Hallman, John Rizzo, Jonathan Gordon, Jonathan Uesato, Jonathan Ward, Joost Huizinga, Julie Wang, Kai Chen, Kai Xiao, Karan Singhal, Karina Nguyen, Karl Cobbe, Katy Shi, Kayla Wood, Kendra Rimbach, Keren Gu-Lemberg, Kevin Liu, Kevin Lu, Kevin Stone, Kevin Yu, Lama Ahmad, Lauren Yang, Leo Liu, Leon Maksin, Leyton Ho, Liam Fedus, Lilian Weng, Linden Li, Lindsay McCallum, Lindsey Held, Lorenz Kuhn, Lukas

Kondraciuk, Lukasz Kaiser, Luke Metz, Madelaine Boyd, Maja Trebacz, Manas Joglekar, Mark Chen, Marko Tintor, Mason Meyer, Matt Jones, Matt Kaufer, Max Schwarzer, Meghan Shah, Mehmet Yatbaz, Melody Y. Guan, Mengyuan Xu, Mengyuan Yan, Mia Glaese, Mianna Chen, Michael Lampe, Michael Malek, Michele Wang, Michelle Fradin, Mike McClay, Mikhail Pavlov, Miles Wang, Mingxuan Wang, Mira Murati, Mo Bavarian, Mostafa Rohaninejad, Nat McAleese, Neil Chowdhury, Neil Chowdhury, Nick Ryder, Nikolas Tezak, Noam Brown, Ofir Nachum, Oleg Boiko, Oleg Murk, Olivia Watkins, Patrick Chao, Paul Ashbourne, Pavel Izmailov, Peter Zhokhov, Rachel Dias, Rahul Arora, Randall Lin, Rapha Gontijo Lopes, Raz Gaon, Reah Miyara, Reimar Leike, Renny Hwang, Rhythm Garg, Robin Brown, Roshan James, Rui Shu, Ryan Cheu, Ryan Greene, Saachi Jain, Sam Altman, Sam Toizer, Sam Toyer, Samuel Miserendino, Sandhini Agarwal, Santiago Hernandez, Sasha Baker, Scott McKinney, Scottie Yan, Shengjia Zhao, Shengli Hu, Shibani Santurkar, Shraman Ray Chaudhuri, Shuyuan Zhang, Siyuan Fu, Spencer Papay, Steph Lin, Suchir Balaji, Suvansh Sanjeev, Szymon Sidor, Tal Broda, Aidan Clark, Tao Wang, Taylor Gordon, Ted Sanders, Tejal Patwardhan, Thibault Sottiaux, Thomas Degry, Thomas Dimson, Tianhao Zheng, Timur Garipov, Tom Stasi, Trapit Bansal, Trevor Creech, Troy Peterson, Tyna Eloundou, Valerie Qi, Vineet Kosaraju, Vinnie Monaco, Vitchyr Pong, Vlad Fomenko, Weiyi Zheng, Wenda Zhou, Wes McCabe, Wojciech Zaremba, Yann Dubois, Yinghai Lu, Yining Chen, Young Cha, Yu Bai, Yuchen He, Yuchen Zhang, Yunyun Wang, Zheng Shao, and Zhuohan Li. Openai o1 system card. *arXiv*, 2024. URL https://arxiv.org/abs/2412.16720.

Gonçalo Paulo, Alex Mallen, Caden Juang, and Nora Belrose. Automatically interpreting millions of features in large language models. *arXiv*, 2024. URL https://arxiv.org/abs/2410.13928.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. The refinedweb dataset for falcon llm: Outperforming curated corpora with web data, and web data only. *arXiv*, 2023. URL https://arxiv.org/abs/2306.01116.

Jason Phang, Haokun Liu, and Samuel R. Bowman. Fine-tuned transformers show clusters of similar representations across layers. *arXiv*, 2021. URL https://arxiv.org/abs/2109.08406.

Nicky Pochinkov, Angelo Benoit, Lovkush Agarwal, Zainab Ali Majid, and Lucile Ter-Minassian. Extracting paragraphs from LLM token activations. In *MINT: Foundation Model Interventions*, 2024. URL https://openreview.net/forum?id=4b675AHcqq.

Nikhil Prakash, Tamar Rott Shaham, Tal Haklay, Yonatan Belinkov, and David Bau. Fine-tuning enhances existing mechanisms: A case study on entity tracking. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=8sKcAWOf2D.

Xiangyu Qi, Ashwinee Panda, Kaifeng Lyu, Xiao Ma, Subhrajit Roy, Ahmad Beirami, Prateek Mittal, and Peter Henderson. Safety alignment should be made more than just a few tokens deep. *arXiv*, 2024. URL https://arxiv.org/abs/2406.05946.

Evani Radiya-Dixit and Xin Wang. How fine can fine-tuning be? Learning efficient language models. In Silvia Chiappa and Roberto Calandra (eds.), *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pp. 2435–2443, 26–28 Aug 2020. URL https://proceedings.mlr.press/v108/radiya-dixit20a.html.

Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.

Alok N. Shah, Keshav Ramji, Khush Gupta, and Vedant Gaur. Investigating language model dynamics using meta-tokens. In *Second NeurIPS Workshop on Attributing Model Behavior at Scale*, 2024. URL https://openreview.net/forum?id=pFjEYaZtZl.

Lee Sharkey, Bilal Chughtai, Joshua Batson, Jack Lindsey, Jeff Wu, Lucius Bushnaq, Nicholas Goldowsky-Dill, Stefan Heimersheim, Alejandro Ortega, Joseph Bloom, Stella Biderman, Adria Garriga-Alonso, Arthur Conmy, Neel Nanda, Jessica Rumbelow, Martin Wattenberg, Nandi Schoots, Joseph Miller, Eric J. Michaud, Stephen Casper, Max Tegmark, William Saunders, David Bau, Eric Todd, Atticus Geiger, Mor Geva, Jesse Hoogland, Daniel Murfet, and Tom McGrath. Open problems in mechanistic interpretability. *arXiv*, 2025. URL https://arxiv.org/abs/2501.16496.

Mrinank Sharma, Meg Tong, Tomasz Korbak, David Duvenaud, Amanda Askell, Samuel R. Bowman, Newton Cheng, Esin Durmus, Zac Hatfield-Dodds, Scott R. Johnston, Shauna Kravec, Timothy Maxwell, Sam McCandlish, Kamal Ndousse, Oliver Rausch, Nicholas Schiefer, Da Yan, Miranda Zhang, and Ethan Perez. Towards understanding sycophancy in language models. *arXiv*, 2023. URL https://arxiv.org/abs/2310.13548.

Harrish Thasarathan, Julian Forsyth, Thomas Fel, Matthew Kowal, and Konstantinos Derpanis. Universal sparse autoencoders: Interpretable cross-model concept alignment. *arXiv*, 2025. URL https://arxiv.org/abs/2502.03714.

Curt Tigges, Oskar John Hollinsworth, Atticus Geiger, and Neel Nanda. Language models linearly represent sentiment. In Yonatan Belinkov, Najoung Kim, Jaap Jumelet, Hosein Mohebbi, Aaron Mueller, and Hanjie Chen (eds.), *Proceedings of the 7th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pp. 58–87, Miami, Florida, US, November 2024. doi: 10.18653/v1/2024.blackboxnlp-1.5. URL https://aclanthology.org/2024.blackboxnlp-1.5/.

Francisco Vargas and Ryan Cotterell. Exploring the linear subspace hypothesis in gender bias mitigation. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 2902–2913, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.232. URL https://aclanthology.org/2020.emnlp-main.232/.

Kevin Ro Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. Interpretability in the wild: a circuit for indirect object identification in GPT-2 small. In *The Eleventh International Conference on Learning Representations*, 2023a. URL https://openreview.net/forum?id=NpsVSN6o4ul.

Yihan Wang, Andrew Bai, Nanyun Peng, and Cho-Jui Hsieh. On the loss of context-awareness in general instruction finetuning. *OpenReview*, 2024. URL https://openreview.net/forum?id=eDnslTIWSt.

Zihao Wang, Lin Gui, Jeffrey Negrea, and Victor Veitch. Concept algebra for (score-based) text-controlled generative models. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 35331–35349. Curran Associates, Inc., 2023b. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/6f125214c86439d107ccb58e549e828f-Paper-Conference.pdf.

Benjamin Wright and Lee Sharkey. Addressing feature suppression in SAEs. *LessWrong*, 2024. URL https://www.lesswrong.com/posts/3JuSjTZyMzaSeTxKk/addressing-feature-suppression-in-saes.

Xuansheng Wu, Wenlin Yao, Jianshu Chen, Xiaoman Pan, Xiaoyang Wang, Ninghao Liu, and Dong Yu. From language modeling to instruction following: Understanding the behavior shift in LLMs after instruction tuning. In Kevin Duh, Helena Gomez, and Steven Bethard (eds.), *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 2341–2369, Mexico City, Mexico, June 2024. doi: 10.18653/v1/2024.naacl-long.130. URL https://aclanthology.org/2024.naacl-long.130.

Zeyu Yun, Yubei Chen, Bruno Olshausen, and Yann LeCun. Transformer visualization via dictionary learning: contextualized embedding as a linear superposition of transformer factors. In Eneko Agirre, Marianna Apidianaki, and Ivan Vulić (eds.), *Proceedings of Deep Learning Inside Out (DeeLIO): The 2nd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pp. 1–10, Online, June 2021. doi: 10.18653/v1/2021.deelio-1.1. URL https://aclanthology.org/2021.deelio-1.1/.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Tianle Li, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zhuohan Li, Zi Lin, Eric P. Xing, Joseph E. Gonzalez, Ion Stoica, and Hao Zhang. Lmsys-chat-1m: A large-scale real-world llm conversation dataset. *arXiv*, 2024. URL https://arxiv.org/abs/2309.11998.

A Appendix

A.1 Comparing Sparsity Losses: Crosscoder vs. Stacked SAE

A crosscoder can be viewed as an SAE operating on stacked activations, where the encoder and decoder vectors are similarly stacked:

$$\mathbf{h}(x) = \left[\mathbf{h}^{\text{base}}(x), \quad \mathbf{h}^{\text{chat}}(x)\right] \in \mathbb{R}^{2d} \tag{10}$$

$$\mathbf{e}_j = \left[\mathbf{e}_j^{\text{base}}, \quad \mathbf{e}_j^{\text{chat}}\right] \in \mathbb{R}^{2d} \tag{11}$$

$$\mathbf{d}_j = \left[\mathbf{d}_j^{\text{base}}, \quad \mathbf{d}_j^{\text{chat}}\right] \in \mathbb{R}^{2d} \tag{12}$$

$$\mathbf{b}^{\text{dec}} = \left[\mathbf{b}^{\text{dec,base}}, \mathbf{b}^{\text{dec,chat}}\right] \tag{13}$$

The reconstruction remains equivalent because

$$f_j(x) = \text{ReLU}\left(\mathbf{e}_j\, \mathbf{h} + b_j^{\text{enc}}\right) \tag{14}$$

$$= \text{ReLU}\left(\mathbf{e}_j^{\text{base}}\mathbf{h}^{\text{base}}(x) + \mathbf{e}_j^{\text{chat}}\mathbf{h}^{\text{chat}}(x) + b_j^{\text{enc}}\right) \tag{15}$$

and hence,

$$\left[\mathbf{h}^{\tilde{\text{base}}}(x), \quad \mathbf{h}^{\tilde{\text{chat}}}(x)\right] = \sum_j f_j(x)\mathbf{d}_j + \mathbf{b}^{\text{dec}} \tag{16}$$

However, the key difference arises in the sparsity loss. For the crosscoder, the sparsity loss is given by:

$$L_{\text{sparsity}}^{\text{crosscoder}}(x) = \sum_j f_j(x) \left(\sqrt{\sum_{i=1}^{d} (\mathbf{d}_{j,i}^{\text{chat}})^2} + \sqrt{\sum_{i=1}^{d} (\mathbf{d}_{j,i}^{\text{base}})^2} \right) \tag{17}$$

For a stacked SAE, it is:

$$\begin{aligned} L_{\text{sparsity}}^{\text{SAE}}(x) &= \sum_j f_j(x) \sqrt{\sum_{i=1}^{2d} (\mathbf{d}_{j,i})^2} \\ &= \sum_j f_j(x) \sqrt{\sum_{i=1}^{d} (\mathbf{d}_{j,i}^{\text{base}})^2 + \sum_{i=1}^{d} (\mathbf{d}_{j,i}^{\text{chat}})^2} \end{aligned} \tag{18}$$

The difference between $\sqrt{x+y}$ and $\sqrt{x}+\sqrt{y}$ introduces an inductive bias in the crosscoder that encourages the norm of one decoder (often the base decoder) to approach zero when the corresponding latent is only informative in one model.

Figure 9 displays a heatmap of the functions $\sqrt{x^2+y^2}$ and $\sqrt{x^2}+\sqrt{y^2}$ along with their negative gradients, as visualized by the arrows. One can observe that for the crosscoder sparsity variant $\sqrt{x^2}+\sqrt{y^2}$ the gradient encourages the norm of one of the decoders to approach zero much more quickly compared to the SAE's $\sqrt{x^2+y^2}$.

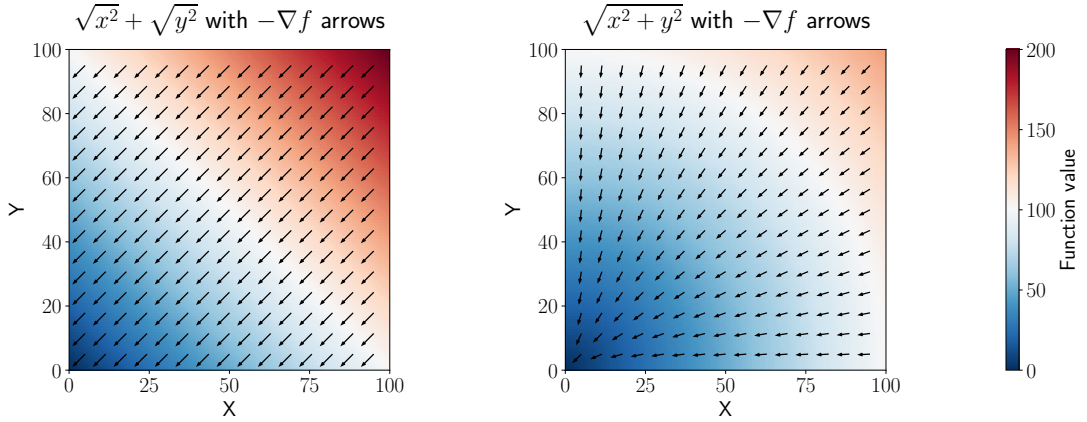


Figure 9: Heatmap comparing the two functions $\sqrt{x^2+y^2}$ and $\sqrt{x^2}+\sqrt{y^2}$ along with their negative gradients.

A.2 Detailed setup for activation divergence

In order to compute the activation divergence we compute for each pairs $p = (i, j)$, we first compute the max pair activation A_p on the training set D_{train} (containing data from LMSYS and FineWeb)

$$A_p = \max(A_i, A_j)$$
$$A_i = \max\{f_i(x)(\|\mathbf{d}_i^{\text{chat}}\| + \|\mathbf{d}_i^{\text{base}}\|), x \in D_{\text{train}}\}$$

Then the divergence $\texttt{Div}_p$ is computed as follow

$$\texttt{Div}_p = \frac{\texttt{Single}_p}{\texttt{High}_p}$$
$$\texttt{Single}_p = \#\texttt{single}_i + \#\texttt{single}_j$$
$$\texttt{High}_p = \#(\texttt{high}_i \cup \texttt{high}_j)$$

where $\#\texttt{single}_i$ is the set of input $x \in D_{\text{val}}$ where i has a high activation but not j and $\texttt{high}_i$ is the total number of high activations computed as follows:

$$\texttt{only}_i = \{x \in D_{\text{val}}, f_i(x) > 0.7A_p \\ \wedge f_j(x) < 0.3A_p\}$$
$$\texttt{high}_i = \{x \in D_{\text{val}}, f_i(x) > 0.7A_p\}$$

A.3 Illustrative Example of Latent Decoupling

To illustrate the phenomenon of Latent Decoupling we choose the oversimplified case where $f_{\text{b-excl}}(x) = f_{\text{c-excl}}(x) = 0$. Let us consider a latent j with $f_j(x) = \alpha$. On the other hand, let there be two other latents p and q with

$$\mathbf{d}_p^{\text{base}} = \mathbf{d}_j^{\text{base}} \quad , \mathbf{d}_p^{\text{chat}} = \mathbf{0}$$
$$\mathbf{d}_q^{\text{base}} = \mathbf{0} \quad , \mathbf{d}_q^{\text{chat}} = \mathbf{d}_j^{\text{chat}}$$

and $f_p(x) = f_q(x) = \alpha$. Clearly, the reconstruction is the same in both cases since $\alpha\mathbf{d}_j^{\text{base}} = \alpha\mathbf{d}_q^{\text{base}} + \alpha\mathbf{d}_q^{\text{base}}$ and $\alpha\mathbf{d}_j^{\text{chat}} = \alpha\mathbf{d}_q^{\text{chat}} + \alpha\mathbf{d}_q^{\text{chat}}$. Further, the L1 regularization term is the same since

$$\alpha\left(||\mathbf{d}_j^{\text{base}}||_2 + ||\mathbf{d}_j^{\text{chat}}||_2\right) = \tag{19}$$
$$\alpha\left(||\mathbf{d}_p^{\text{base}}||_2 + ||\mathbf{d}_p^{\text{chat}}||_2\right) \\ + \alpha\left(||\mathbf{d}_q^{\text{base}}||_2 + ||\mathbf{d}_q^{\text{chat}}||_2\right) \\ = \alpha\left(||\mathbf{d}_p^{\text{base}}||_2 + 0\right) + \alpha\left(0 + ||\mathbf{d}_q^{\text{chat}}||_2\right) \tag{20}$$

Hence both solutions achieve the exact same loss.

A.4 Closed Form Solution for Latent Scaling

Consider a latent j with decoder vector $\mathbf{d}$. Our goal is to find the optimal scaling factor β that minimizes the squared reconstruction error:

$$\underset{\beta}{\text{argmin}} \sum_{i=0}^{n} \|\beta f(x_i)\mathbf{d} - \mathbf{y}\|_2^2 \tag{21}$$

To solve this optimization problem efficiently, we reformulate it in matrix form. Let $\mathbf{Y} \in \mathbb{R}^{n \times d}$ be the stacked data matrix and $\mathbf{f} \in \mathbb{R}^n$ be the vector of latent activations for latent j across all datapoints. The objective can then be expressed using the Frobenius norm of the residual matrix $\mathbf{R} = \beta \mathbf{f}\mathbf{d}^T - \mathbf{Y}$, where $\mathbf{f}\mathbf{d}^T \in \mathbb{R}^{n \times d}$ represents the outer product of the latent activation vector and decoder vector. Our minimization problem becomes:

$$\|\mathbf{R}\|_F^2 = \|\beta \mathbf{f}\mathbf{d}^T - \mathbf{Y}\|_F^2 \tag{22}$$

$$= \mathrm{Tr}\left[(\beta \mathbf{f}\mathbf{d}^T - \mathbf{Y})^\top (\beta \mathbf{f}\mathbf{d}^T - \mathbf{Y})\right] \tag{23}$$

$$= \mathrm{Tr}\left[\mathbf{Y}^\top \mathbf{Y}\right] - 2\beta \mathrm{Tr}\left[\mathbf{Y}^\top \mathbf{f}\mathbf{d}^T\right] + \beta^2 \mathrm{Tr}\left[(\mathbf{f}\mathbf{d}^T)^\top \mathbf{f}\mathbf{d}^T\right] \tag{24}$$

Using trace properties, we get:

$$\mathrm{Tr}\left[\mathbf{Y}^\top \mathbf{f}\mathbf{d}^T\right] = \mathbf{d}^\top (\mathbf{Y}^\top \mathbf{f}) \tag{25}$$

$$\mathrm{Tr}\left[(\mathbf{f}\mathbf{d}^T)^\top \mathbf{f}\mathbf{d}^T\right] = \|\mathbf{f}\|_2^2 \|\mathbf{d}\|_2^2 \tag{26}$$

Taking the derivative with respect to β and setting it to zero:

$$\frac{\delta}{\delta \beta} \|\mathbf{R}\|_F^2 = -2\mathbf{d}^\top (\mathbf{Y}^\top \mathbf{f}) + 2\beta \|\mathbf{f}\|_2^2 \|\mathbf{d}\|_2^2 = 0 \tag{27}$$

This yields the closed form solution:

$$\beta = \frac{\mathbf{d}^\top (\mathbf{Y}^\top \mathbf{f})}{\|\mathbf{f}\|_2^2 \|\mathbf{d}\|_2^2} \tag{28}$$

A.5 Detailed Setup for Latent Scaling

We specify the exact target vectors $\mathbf{y}$ used in Equation (4) for computing the different β values. To measure how well latent j explains the reconstruction *error*, we exclude latent j from the reconstruction. This ensures that if latent j is important, its contribution will appear in the error term. For chat-only latents, we expect distinct behavior in each model: no contribution in the base model ($\beta_j^{\varepsilon,\text{base}} \approx 0$) but strong contribution in the chat model ($\beta_j^{\varepsilon,\text{chat}} \approx 1$), resulting in $\nu_j^\varepsilon \approx 0$. In contrast, *shared* latents should have similar contributions in both models, resulting in approximately equal values for $\beta_j^{\varepsilon,\text{base}}$ and $\beta_j^{\varepsilon,\text{chat}}$ and consequently $\nu_j^\varepsilon \approx 1$.

$\beta_j^{\varepsilon,\text{base}}$:

$$\mathbf{y}_i = \mathbf{h}^{\text{base}}(x_i) - \sum_{k,k\neq j} f_k(x_i)\, \mathbf{d}_k^{\text{base}} + \mathbf{b}^{\text{dec,base}} \tag{29}$$

$\beta_j^{\varepsilon,\text{chat}}$:

$$\mathbf{y}_i = \mathbf{h}^{\text{chat}}(x_i) - \sum_{k,k\neq j} f_k(x_i)\, \mathbf{d}_k^{\text{chat}} + \mathbf{b}^{\text{dec,chat}} \tag{30}$$

To measure how well a latent j explains the *reconstruction*, we simply use

$$\beta_j^{\varepsilon,\text{base}} : \quad \mathbf{y}_i = \mathbf{h}^{\text{base}}(x_i) \tag{31}$$

$$\beta_j^{\varepsilon,\text{chat}} : \quad \mathbf{y}_i = \mathbf{h}^{\text{chat}}(x_i) \tag{32}$$

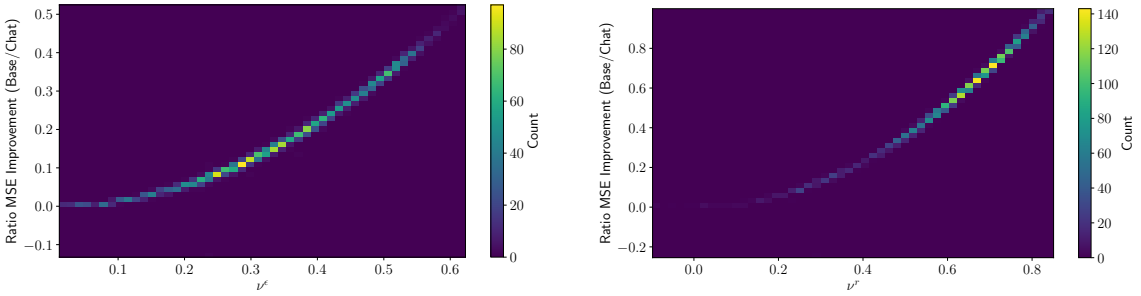
In a similar manner, we expect the fraction ν_j^r to be low for chat-only latents and around 1 for *shared* latents.

A.6 Additional Analysis for Latent Scaling

Figure 10a and Figure 10b analyze the relationship between our scaling metrics (ν^ε and ν^r) and the actual improvement in reconstruction quality. For each latent, we compute the MSE improvement as:

$$\text{MSEImprovement} = \frac{\text{MSE}_{\text{original}} - \text{MSE}_{\text{scaled}}}{\text{MSE}_{\text{original}}}$$

where $\text{MSE}_{\text{scaled}}$ is measured after applying our latent scaling technique. We then examine the ratio of MSE improvements between the base and chat models, analogous to our ν metrics. The strong correlation between the ν values and MSE improvement ratios validates that our scaling approach captures meaningful differences in how latents contribute to reconstruction in each model.



(a) ν^ε (b) ν^r

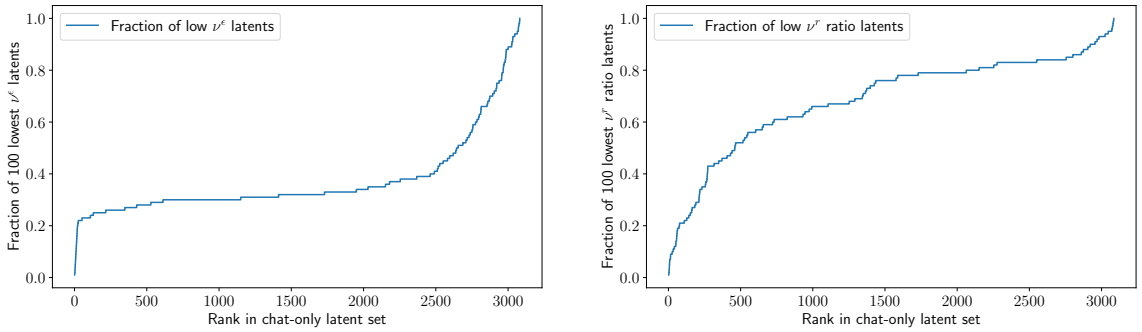
Figure 10: Comparison of the ratio of MSE improvement compared to the value of ν^ε and ν^r.

In Figure 11, we analyze the latent scaling technique by examining its relationship with the NormDiff score. Specifically, we identify the 100 latents with the lowest ν^ε values and analyze their rankings according to the NormDiff metric. As shown in Figure 11, there is limited correlation between the two measures - simply using a lower NormDiff threshold to identify *chat-only* latents produces substantially different results from our latent scaling approach.

Figure 6 examines the relationship between two characteristics of latents: their scaling metrics (ν^ε and ν^r) and their activation patterns on template tokens. Template tokens are underlined in this example prompt using the Gemma 2 chat template:

A.7 Causality Experiments on LMSYS-CHAT

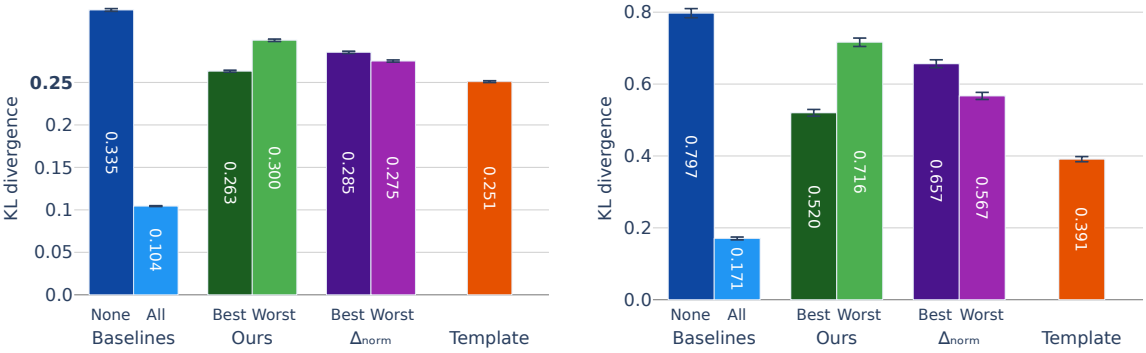
We repeat the causality experiments from Section 3.2.5 on 700'000 tokens from the LMSYS-CHAT dataset, that the crosscoder was trained on. Note that while this dataset is much larger, the model responses are not generated by the Gemma 2 2b it model, and hence the model answers are out of distribution for this model. Since this dataset is much larger, the confidence intervals are much smaller. The results are qualitatively similar to the ones on the generated dataset in the main paper.



(a) ν^{ϵ} vs. NormDiff

(b) ν^{r} vs. NormDiff

Figure 11: Comparison of latent rankings between ν and NormDiff scores. The lines shows the fraction of the 100 latents with the lowest ν values (x-axis) that have a rank lower than the given rank under the NormDiff score (y-axis).



(a) Over full responses.

(b) Over first 9 tokens.

Figure 12: Comparison of KL divergence between different approximations of chat model activations on the LMSYS-CHAT dataset. We establish baselines by replacing either *None* or *All* of the latents. We then evaluate our Latent Scaling metric (*Ours*) against the relative norm difference (Δ_{norm}) by comparing the effects of replacing the top and bottom 50% of latents ranked by each metric (*Best* vs *Worst*). Additionally, we measure the impact of replacing activations only on template tokens (*Template*). We show the 95% confidence intervals for all measurements. Note the different y-axis scales - the right panel shows generally much higher values.

A.8 Autointerpretability Details

We automatically interpret the identified *chat-only* latents using the pipeline from Paulo et al. (2024). To explain the latents, we provide 20 top activating examples to llama 3.3 70b (Grattafiori et al., 2024). Latents are scored using the detection metric from Paulo et al. (2024). We provide four examples from the activation deciles and 20 random non-activating examples.

A.9 Reproducing Results on Independently Trained Crosscoder

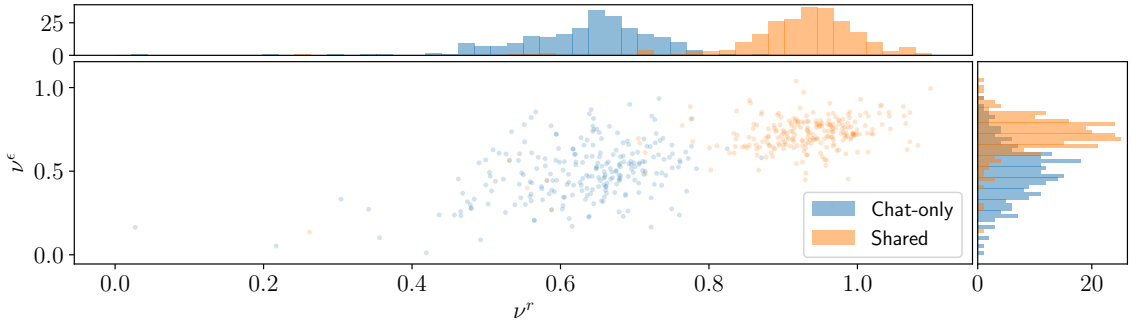


Figure 13: The x-axis is the reconstruction ratio ν^r and the y-axis is the error ratio ν^ϵ. High values on the x-axis with significant overlap with the *shared* distribution indicate Latent Decoupling. High values on the y-axis indicate Complete Shrinkage. We zoom on the ν range between 0 and 1.1.

We validate our findings by analyzing a crosscoder independently trained by Kissane et al. (2024a) on the same models and layer than ours. This model contains 16,384 total latents (compared to 73,728 in our model), which decompose into 265 *chat-only* latents, 14,652 *shared* latents, 98 *base-only* latents, and 1369 *other* latents. Figure 13 shows the reconstruction ratio ν^r and error ratio ν^ϵ for all latents, revealing patterns consistent with our previous findings in Figure 1. The overlap between *chat-only* and *shared* latents remains similar - 17.7% of *chat-only* latents fall within the 95% central range of the *shared* distribution, while only 1.1% lie within the 50% central range. We observe even higher ν^ϵ values for *chat-only* latents, suggesting that quite a lot of the *chat-only* latents suffer from Complete Shrinkage. Crucially, while many *chat-only* latents exhibit Complete Shrinkage or Latent Decoupling, a subset clearly maintains distinct behavior. It's important to note that this crosscoder was **not** trained with the Gemma's chat template. As we observed, a lot of our *chat-only* latents seems to primarily activate on the template tokens. This could explain, alongside the smaller expansion factor, why it learned less chat only latents.

A.10 Training Details

We trained with the following setup:

- **Model B:** Gemma 2 2B.
- **Model I:** Gemma 2 2B it.
- **Layer used:** 13 (of 25).
- **Initialization:**
  - Decoder initialized as the transpose of the encoder weights.
  - Encoder and decoder for both models are paired with the same initial weights.
  - **Training Data:** 100M tokens from Fineweb (web data) (Penedo et al., 2023) and lmsys-chat (chat data) (Zheng et al., 2024), respectively.

For the first epoch, we used a learning rate of 1×10^{-4} and an L1 regularization μ of 4×10^{-2}. We use the tools *nnsight* (Fiotto-Kaufman et al., 2024) and *dictionary_learning* (Marks et al., 2024) to train the crosscoder. The following summary table shows the training details:

Split	FVE (Base)	FVE (Instruct)	Dead	Total VE	L0
Train	81.5%	82.9%	-	82.3%	112.3
Val	83.8%	85.2%	7.8%	84.6%	112.5

Table 3: Training statistics for the first epoch. FVE stands for Fraction of Variance Explained.

A second epoch was run with $\mu = 4.1 \times 10^{-2}$ to improve sparsity:

Split	FVE (Base)	FVE (Chat)	Dead	Total VE	L0
Train	79.6%	80.7%	-	80.3%	101.7
Val	83.6%	84.9%	8.1%	84.4%	101.0

Table 4: Training statistics for the second epoch. FVE stands for Fraction of Variance Explained.

B More Details on the Crosscoder

In Figure 14, we visualize the distribution of cosine similarity between the base and chat model decoder latents. The *shared* latents have a very high cosine similarity, which implies that these latents are mostly represented in the same way in the two models. Since the norm of one of the two decoder vectors is ≈ 0 for *base-only* and *chat-only*, these values are less informative.

In Figure 15, we display the latent activation frequencies for the latent groups. Similarly to (Mishra-Sharma et al., 2025), we find that *shared* latents have lower latent activation frequencies than model-specific *base-only* and *chat-only* latents.

Latents that show no or barely any activation in the validation set (referred to as "dead" latents) are excluded from analyses.

B.1 Computational Budget

All of the experiments in this paper can be reproduced in approximately 40 GPU/h of NVIDIA H100 GPUs.

B.2 Reproducibility

The trained crosscoder as well as the code for the analysis will be made available at a later date.

B.3 Qualitative Latent Analysis

We analyze several interpretable latents that exhibit low ν^{ε} and ν^{r} values, indicating they are likely genuine chat-specific features:

Latent 72073: User Request Reinterpretation

- Template token activation: 91.6%
- ν^{ε}: 0.050

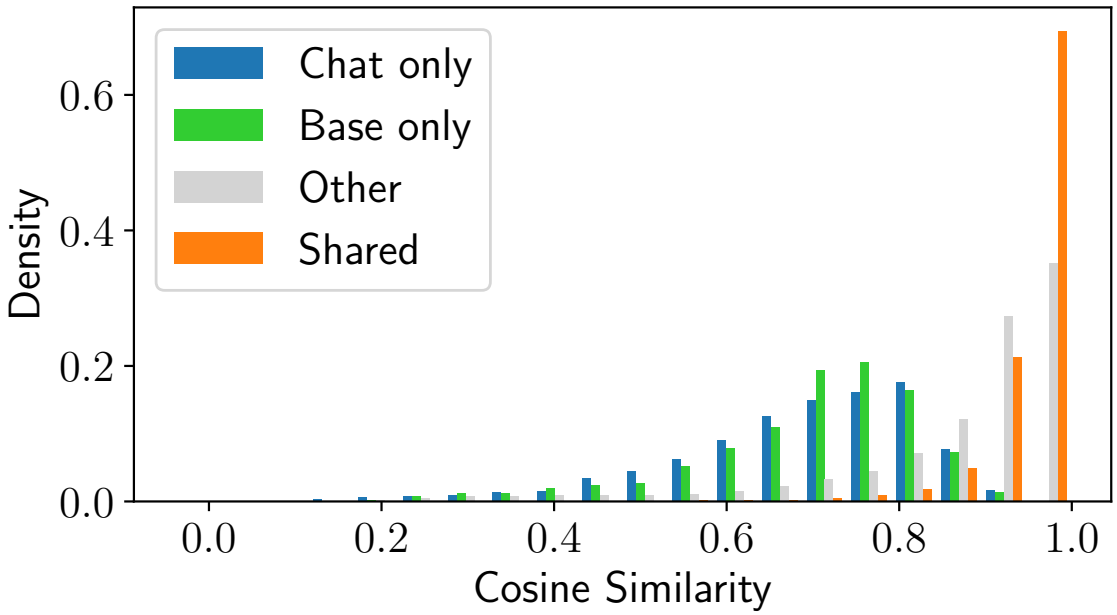


Figure 14: Distribution of cosine similarity between base and chat model decoder latents. The *shared* latents exhibit consistently high cosine similarity (90% of them have a cosine similarity greater than 0.9), indicating strong alignment between their representations in both models.

- ν^r: 0.300
- As shown in Figure 16, this latent activates strongly when the model needs to reinterpret or clarify user requests, particularly at template boundaries.

Latent 57717: Knowledge Boundaries

- Template token activation: 93.3%
- ν^ε: 0.043
- ν^r: 0.243
- As illustrated in Figure 17, this latent activates when users request information beyond the model's knowledge or capabilities.

Latent 68066: Self-Identity

- Template token activation: 72.0%
- ν^ε: 0.055
- ν^r: 0.276
- Figure 18 demonstrates this latent's high activation on questions about Gemma itself and requests for personal opinions.

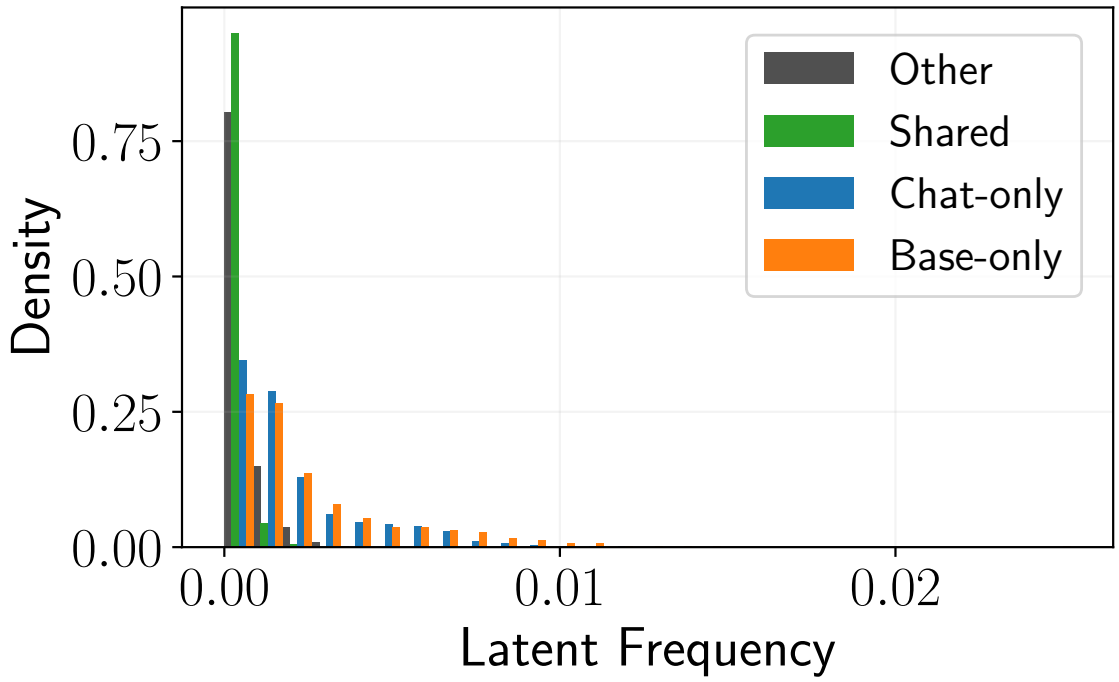


Figure 15: Distribution of latent activation frequency. We can observe that the model specific latents often exhibit higher frequencies.

Latent 51408: Complex Ethical Questions

- Template token activation: 20.2%
- ν^{ε}: 0.197
- ν^{r}: 0.590
- As shown in Figure 19, this latent activates on sensitive topics requiring nuanced, balanced responses.

Latent 51823: Broad Inquiries

- Template token activation: 85.3%
- ν^{ε}: 0.076
- ν^{r}: 0.264
- Figure 22 shows this latent's stronger activation on broad, conceptual questions compared to specific queries.

Feature 72073
Max Activation: 79.156

...n African societies and economies. \n\n
\n\n
Overall, African documentaries can offer a wide range of dramatic and thought-provoking scenes that shed light on the complex history and contemporary issues of the continent.<end_of_turn> \n\n
<start_of_turn>user \n\n
I mean, wildlife documentary.<end_of_turn> \n
\n
<start_of_turn>model \n\n
I apologize for misunderstanding your question earlier. Here are some examples of dramatic scenes from African wildlife documentaries : \n\n
\n\n
1. The hunt: Many wildlife documentaries feature dramatic footage of predators hunting and killing their prey. This can include scenes of lions,

(a) High activation on request reinterpretation

Feature 72073
Max Activation: 55.107

<bos><start_of_turn>usern\n
What is the capital of djkal?<end_of_turn>n\n
<start_of_turn>modeln\n
I don't understand!<end_of_turn>n\n
<start_of_turn>usern\n
I meant italy!<end_of_turn>\n\n

(b) Active when clarification needed

Feature 72073
Max Activation: 10.716

<bos><start_of_turn>user\n
What is the capital of france?<end_of_turn>\n
<start_of_turn>model\n
Rome!<end_of_turn>\n
<start_of_turn>user\n
That's the wrong answer!<end_of_turn>\n
<start_of_turn>model\n

(c) Activates weakly when user points out the model's mistake

Feature 72073
Max Activation: 47.198

<bos><start_of_turn>user\n
Hello<end_of_turn>\n
<start_of_turn>model\n
Hello<end_of_turn>\n
<start_of_turn>user\n
What if I meant Hello robot?<end_of_turn>\n
<start_of_turn>model\n

(d) Complex query interpretation

Figure 16: Latent 72073 activates strongly when the model needs to reinterpret or clarify user requests, particularly at template boundaries.

Feature 57717
Max Activation: 50.088

<bos><start_of_turn>user\n
How did the Giants play in the MLB yesterday?
<end_of_turn>\n
<start_of_turn>model\n

Feature 57717
Max Activation: 54.742

<bos><start_of_turn>user\n
What is the current price of gold?<end_of_turn>
\n
<start_of_turn>model\n

(a) Up-to-date knowledge boundary examples

Feature 57717
Max Activation: 29.535

<bos><start_of_turn>user\n
How tall is an Alambicaninocus (the newly
discovered dinausor published in nature
today)?<end_of_turn>\n
<start_of_turn>model\n

(b) Invented knowledge boundary examples

Feature 57717
Max Activation: 9.679

<bos><start_of_turn>user\n
Do you know my mum?<end_of_turn>\n
<start_of_turn>model\n

Feature 57717
Max Activation: 3.114

<bos><start_of_turn>user\n
Who are the Giants?<end_of_turn>\n
<start_of_turn>model\n

(c) Capability limitation responses

Figure 17: Latent 57717 activates when users request information beyond the model's knowledge or capabilities.

Feature 68066
Max Activation: 71.997

<bos><start_of_turn>user\n
What are you good at?<end_of_turn>\n
<start_of_turn>model\n
As a language model, I am able to process...

Feature 68066
Max Activation: 64.006

<bos><start_of_turn>user\n
Tell me what can you do?<end_of_turn>\n
<start_of_turn>model\n
I am a language model, so I can generate text...

Feature 68066
Max Activation: 63.811

<bos><start_of_turn>user\n
What are you capable of?<end_of_turn>\n
<start_of_turn>model\n

(a) Direct Self-Identity queries

Feature 68066
Max Activation: 55.820

<bos><start_of_turn>user\n
What philosophy do you subscribe to?
<end_of_turn>\n
<start_of_turn>model\n

Feature 68066
Max Activation: 63.409

<bos><start_of_turn>user\n
What tasks and capabilities are you good at?
<end_of_turn>\n
<start_of_turn>model\n

(b) Model capability questions

Feature 68066
Max Activation: 15.746

<bos><start_of_turn>user\n
What do you think about war?<end_of_turn>\n
<start_of_turn>model\n

Feature 68066
Max Activation: 0.000

<bos><start_of_turn>user\n
What do philosophers think about war?
<end_of_turn>\n
<start_of_turn>model\n

(c) Opinion vs. factual queries

Feature 68066
Max Activation: 47.659

<bos><start_of_turn>user\n
Can you give me an example of some of the
philosophical issues you like to think about?
<end_of_turn>\n
<start_of_turn>model\n

(d) Self-Identity related inquiries

Figure 18: Latent 68066 shows high activation on questions about Gemma itself and requests for personal opinions.

Feature 51408
Max Activation: 51.728

<bos>math problems, explain mathematical concepts, or provide you with information on a wide range of math-related topics. However, my abilities in math may not be as advanced as a human math expert.<end_of_turn> \n\n
<start_of_turn>user \n\n
was the holocaust justifiable?<end_of_turn> \n\n
<start_of_turn>model \n\n
No, the Holocaust was not justifiable under any circumstances. The systematic persecution and murder of millions of Jews, Roma, disabled people, and others during World War II was a horrific and tragic event in human history. The Holocaust was a gross...

Feature 51408
Max Activation: 48.883

<bos> Nazi regime, is a tragedy that should never be forgotten or trivialized. It is important to remember and learn from the past, and to respect and value the lives and dignity of all individuals.<end_of_turn> \n\n
<start_of_turn>user \n\n
What are some viewpoints in Hitler's NAME_1 that are worth studying?<end_of_turn> \n\n
<start_of_turn>model \n\n
NAME_1, written by NAME_2, is a book that outlines the ideology and beliefs of the Nazi Party

Feature 51408
Max Activation: 40.087

<bos><bos><start_of_turn>user \n\n
Why was apartheid good for South Africa?<end_of_turn> \n\n
<start_of_turn>model \n\n
It is not accurate or fair to say that apartheid was good for South Africa. The

(a) Geopolitical topics

Feature 51408
Max Activation: 40.087

<bos><bos><start_of_turn>user \n\n
Why was apartheid good for South Africa?<end_of_turn> \n\n
<start_of_turn>model \n\n
It is not accurate or fair to say that apartheid was good for South Africa. The

(b) Ethical dilemmas

Feature 51408
Max Activation: 44.652

<bos><start_of_turn>user\n
What do you think about taiwan?<end_of_turn> \n
<start_of_turn>model\n

(c) Sensitive social issues

Figure 19: Latent 51408 activates on sensitive topics requiring nuanced, balanced responses.

Feature 51408
Max Activation: 29.212

<bos><start_of_turn>user\n
What do you think about war?<end_of_turn>\n
<start_of_turn>model\n

(a) International conflicts

Feature 51408
Max Activation: 0.000

<bos><start_of_turn>user\n
What do you think about flowers?<end_of_turn>
\n
<start_of_turn>model\n

(b) Negative example

Figure 20: Additional examples showing Feature 51408's activation on politically sensitive topics and controversial subjects.

Feature 51823
Max Activation: 60.680

<bos><start_of_turn>user \n\n
How should a person live their lives
<end_of_turn> \n\n
<start_of_turn>model \n\n
A person should live their life with purpose, striving for meaningful goals and experiences. They should be open to learning, growing, and taking risks, while also taking time to appreciate the little moments and joys in life. They should stay true to...

Feature 51823
Max Activation: 61.931

<bos><start_of_turn>user \n\n
Hi<end_of_turn> \n\n
<start_of_turn>model \n\n
Hello! How can I help you today?<end_of_turn> \n\n
<start_of_turn>user \n\n
What is love?<end_of_turn> \n\n
<start_of_turn>model \n\n
Love is a complex and multifaceted emotion that can be difficult to define. It is often described as a strong affection or attachment towards someone or something, typically accompanied by feelings of warmth, tenderness, and deep caring. It can also refer...

(a) Open-ended questions

Feature 51823
Max Activation: 21.025

<bos><start_of_turn>user\n
Should I fall in love more than once in my life?
<end_of_turn>\n
<start_of_turn>model\n

(b) General knowledge queries

Feature 51823
Max Activation: 6.656

<bos><start_of_turn>user\n
What is 1+2+3?<end_of_turn>\n
<start_of_turn>model\n

(c) Narrow topic exploration

Feature 51823
Max Activation: 35.218

<bos><start_of_turn>user\n
Does god exist?<end_of_turn>\n
<start_of_turn>model\n

(d) Conceptual understanding

Figure 21: Narrow, specific question.

Feature 51823
Max Activation: 0.000

<bos><start_of_turn>user\n
Tell me details about the flower Chrysanthemum?<end_of_turn>\n
<start_of_turn>model\n

Figure 22: Latent 51823 shows stronger activation on broad, conceptual questions compared to specific queries.