Steering Knowledge Selection Behaviours in LLMs via SAE-Based Representation Engineering

Anonymous ACL submission

Abstract

Large language models (LLMs) can store a significant amount of factual knowledge in their parameters. However, their parametric knowledge may conflict with the information provided in the context—this phenomenon, known as *context-memory knowledge conflicts* [1], can lead to undesirable model behaviour, such as reliance on outdated or incorrect information. Analysing the internal activations of LLMs, we find that they can internally register the signals of knowledge conflict at mid-layers. Such signals allow us to detect whether a knowledge conflict occurs and use *inference-time* intervention strategies to resolve it. In this work, we propose SPARE, a *training-free* representation engineering method that uses pre-trained sparse auto-encoders (SAEs) to control the knowledge selection behaviour of LLMs. SPARE identifies the functional features that control the knowledge selection behaviours and applies them to edit the internal activations of LLMs at inference time. Our experimental results show that SPARE can effectively control the usage of either knowledge source to resolve knowledge conflict in open-domain question-answering tasks, surpassing existing representation engineering methods (+10%) as well as contrastive decoding methods (+15%).

1 Introduction

Large language models (LLMs) have shown remarkable capability to memorise factual knowledge and solve knowledge-intensive tasks (Petroni et al., 2019; Brown, 2020; Touvron et al., 2023; Jiang et al., 2023; Anil et al., 2023). Nevertheless, the knowledge stored in their parameters (*parametric knowledge*) can be inaccurate or outdated (Xu et al., 2024). To alleviate this issue, retrieval and tool-augmented approaches have been widely adopted to provide LLMs with external knowledge (*contextual knowledge*) (Karpukhin et al., 2020; Lewis et al., 2020; Wu et al., 2022; Schick et al., 2024). However, contextual knowledge may sometimes conflict with the parametric knowledge of the model, leading to what we refer to as *knowledge conflicts*. Such conflicts can cause undesired behaviour, where the model may rely on inaccurate information sources, resulting in incorrect outputs (Mallen et al., 2023; Xie et al., 2024a; Su et al., 2024; Wang et al., 2023; Zhao et al., 2024).

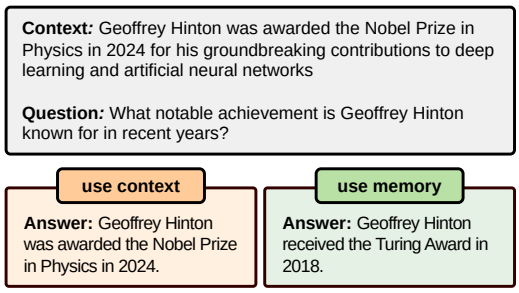


Figure 1: In the event of a knowledge conflict, the model can rely on the context or on the parametric knowledge.

Prior research found that LLMs tend to prefer contextual knowledge (e.g., retrieved passages) over their parametric knowledge when conflicts occur (Su et al., 2024; Xie et al., 2024a; Hong et al., 2024). For instance, Su et al. (2024) show that most LLMs choose parametric knowledge in less than 10% examples. However, in more general applications, LLMs should retain the ability to use their parametric knowledge when presented with misinformation (Chen and Shu, 2023b,a; Zou et al., 2024; Mallen et al., 2023; Zhong et al., 2023). Existing works investigate fine-tuning and prompting-based strategies to detect and resolve knowledge conflicts (Wang et al., 2023); however, they need additional interactions with the model, e.g., by asking the LLMs to examine the conflicts sentence by sentence, resulting in high latency times and preventing practical applications.

In this work, we investigate *representation engineering* methods to efficiently steer the usage

[1] We will refer to these as *knowledge conflicts* for brevity.

of parametric and contextual knowledge of LLMs at *inference time*. Although representation engineering has provided an efficient and transparent framework for controlling the behaviour of LLMs, we find that existing methods fail to effectively steer knowledge usage. This may be because these methods directly modify the internal activations of LLMs, such as hidden states (Turner et al., 2023a; Zou et al., 2023a) or MLP activations (Qiu et al., 2024; Meng et al., 2022). These activations are polysemantic dense vectors that overlap with many independent semantic features (Olah, 2023). Thus, minor edits in one dimension can influence multiple semantic features, making it difficult to adjust activations accurately without affecting other features in practice.

Recently, sparse auto-encoders (SAEs) have been proposed to address this limitation, as they allow to decompose the polysemantic activations of LLMs into a large-scale monosemantic feature dictionary (Huben et al., 2024; Gao et al., 2024; Templeton et al., 2024a). Therefore, we introduce SAEs as a tool for precise activation editing to guide the knowledge selection of LLMs. Specifically, we propose SPARE, a **Sp**arse **A**uto-Encoder-based **R**epresentation **E**ngineering method to steer the knowledge selection behavior of LLMs. SPARE first identifies the SAE activations that are related to specific knowledge selection behaviours (Section 4.2); then, it extracts functional features that control the usage of contextual and parametric knowledge, and finally applies them to steer the behaviour of the model (Section 4.3).

Our experimental results on open-domain question-answering tasks show that SPARE effectively controls the knowledge selection behaviours by modifying a small set of SAE features—less than 0.4% SAE activations for 4 layers. SPARE yields more accurate results than state-of-the-art representation engineering methods ($+10\%$), contrastive decoding ($+15\%$), and in-context learning ($+7\%$), achieving the best performance on steering knowledge selection behaviours of LLMs under knowledge conflicts.

2 Background

Problem Setup Following Longpre et al. (2021); Hong et al. (2024); Xie et al. (2024a), we use open-domain question-answering (ODQA) tasks to investigate the behaviour of LLMs when there is a conflict between the parametric knowledge of the model and contextual knowledge. In ODQA datasets with knowledge conflicts, each instance is presented as (Q, E_M, M, E_C, C), where Q is the question, E_M is the evidence that supports the memorised knowledge stored in the model parameters, E_C is the evidence that conflicts with the language model's memorised knowledge, M is the answer based on the E_M, and C is the answer based on the E_C.

Sparse Auto-Encoders Recent works have proposed using sparse auto-encoders (SAEs) to interpret the complex representations of LLMs by decomposing them into a large set of monosemantic features (Huben et al., 2024; Gao et al., 2024; Templeton et al., 2024a). Given an activation $\mathbf{h} \in \mathbb{R}^d$ from the residual stream of LLMs, a SAE with n latent dimensions encodes it into a sparse vector $\mathbf{z} \in \mathbb{R}^n$ and decodes it to recover $\mathbf{h}$:

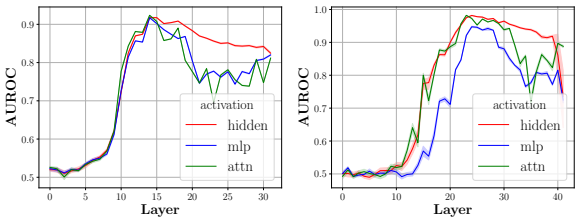
$$f_\theta(\mathbf{h}) = \sigma\left(\mathbf{W}_\theta\left(\mathbf{h} - \mathbf{b}\right) + \mathbf{b}_\theta\right) = \mathbf{z},$$
$$g_\phi(\mathbf{z}) = \mathbf{W}_\phi\,\mathbf{z} = \sum_{i=1}^{n} z_i \mathbf{f}_i + \mathbf{b} = \hat{\mathbf{h}} \quad (1)$$

where σ is an activation function that outputs a non-negative value such as ReLU, $\mathbf{W}_\theta \in \mathbb{R}^{n\times d}$, $\mathbf{b} \in \mathbb{R}^d$, $\mathbf{b}_\theta \in \mathbb{R}^n$, $\mathbf{W}_\phi \in \mathbb{R}^{d\times n}$, z_i is the i-th element of the SAE activation $\mathbf{z}$, and $\mathbf{f}_i \in \mathbb{R}^d$ is the i-th column of $\mathbf{W}_\phi$. The $\{\mathbf{f}_i\}_{i=1}^n$ learned through the SAE are considered highly monosemantic, and the SAE activation $\mathbf{z}$ indicates the activated values of $\{\mathbf{f}_i\}_{i=1}^n$.

3 Detection of Knowledge Conflicts

In this section we investigate whether we can detect the occurrence of conflicts during the generation process, since identifying such conflicts is a prerequisite for exploring inference-time strategies to control the LLM.

We focus on the residual stream (Elhage et al., 2021) of the model and look for a signal of knowledge conflict. To this end, we create two groups of input instances, $\mathcal{D}_{E_M} = \{(Q, E_M)\}$ and $\mathcal{D}_{E_C} = \{(Q, E_C)\}$. In $\mathcal{D}_{E_M}$, the model is provided with a context that is coherent with the model internal memorized knowledge, whereas in $\mathcal{D}_{E_C}$ the model is provided with a context that does not agree with model parametric knowledge, thus causing a knowledge conflict. To determine whether a signal of conflict arises in the residual stream, we focus on the last position of the sequence during generation, which is supposed to encode the information to predict the first token of the answer.



(a) Llama2-7B (b) Gemma2-9B

Figure 2: The knowledge conflict probing results of Llama2-7B and Gemma2-9B on NQSwap (Longpre et al., 2021). The probing results on hidden states, MLP and Self-Attention activations are coloured differently.

We apply a linear probing method (Conneau et al., 2018; Zhu and Li, 2023; Allen-Zhu and Li, 2023) to investigate whether the residual stream contains a signal of knowledge conflict. Specifically, we train logistic regression models to classify whether a given activation (the hidden state, MLP or Self-Attention activations) is from the $\mathcal{D}_{E_C}$ or $\mathcal{D}_{E_M}$, i.e. whether it contains a knowledge conflict or not. We use activations from each layer as input and formulate this as a binary classification task. The evaluation is conducted on a held-out test set. We present probing results on Llama2-7B (Touvron et al., 2023) and Gemma2-9B (Rivière et al., 2024) using AUROC as metric in Fig. 2. We observe that the probing accuracy increases from the first layer to the middle layers, and this trend is the same across different types of activations. This indicates that we can detect the signal of knowledge conflict in the residual stream of the mid-layers. We provide more details and analysis about knowledge conflict detection in Appendix A.

The above analysis shows that knowledge conflicts can be identified in the internal states of LLMs. Moreover, it provides insight into which layers can be more influential in the knowledge selection (Section 5.4).

4 Resolving Knowledge Conflicts by Representation Engineering

In this section, we introduce SPARE, our SAE-based representation engineering method, to control the usage of parametric and contextual knowledge to resolve conflicts. SPARE consists of the three following steps: *1)* collecting activations that lead to different knowledge selection behaviours (Section 4.1); *2)* identifying SAE activations that are related to each knowledge selection behaviour (Section 4.2); *3)* steering the usage of either knowledge source by editing the hidden states of LLMs at inference time (Section 4.3).

4.1 Collecting Activations with Different Knowledge Selection Behaviours

We showed in Section 3 that we can detect the knowledge conflict by probing the residual stream. We now want to characterise the activations that lead to different knowledge selection behaviours. To this end, given a set of instances $\mathcal{D}_{E_C}$ that cause a knowledge conflict, we separate it into two groups based on the model's predictions: $\mathcal{D}_C$, where the model generates an answer that aligns with the context, and $\mathcal{D}_M$, where the model ignores the context and generates an answer relying on the parametric knowledge. These two subsets characterise two knowledge selection behaviours of the model. In the following, we omit the notation to specify the layer of $\mathbf{h}$ and $\mathbf{z}$ for simplicity, as the method can be applied to arbitrary layers.

We collect the hidden states from $\mathcal{D}_C$ and $\mathcal{D}_M$ for N samples, denoting them as $\{\mathbf{h}_C^j\}_{j=1}^N$ and $\{\mathbf{h}_M^j\}_{j=1}^N$, respectively. We then obtain the SAE activation for each sample by $\mathbf{z}_C^j = f_\theta(\mathbf{h}_C^j)$ and $\mathbf{z}_M^j = f_\theta(\mathbf{h}_M^j)$. Finally, we compute the average of the sets $\{\mathbf{z}_C^j\}_{j=1}^N$ and $\{\mathbf{z}_M^j\}_{j=1}^N$ to obtain the mean vectors $\overline{\mathbf{z}_C}$ and $\overline{\mathbf{z}_M}$, respectively. More details on this step are presented in Appendix C.1.

At this stage, $\overline{\mathbf{z}_C}$ and $\overline{\mathbf{z}_M}$ contain the information to steer the generation towards C or M. However, there might still be instance-specific activations with non-zero values that are not responsible for the knowledge selection behaviour. In the next section, we identify functional activations related to knowledge selection behaviours and then construct two orthogonal SAE activations, $\mathbf{z}_C$ and $\mathbf{z}_M$ for steering the knowledge selection behaviours.

4.2 Identifying Functional SAE Activations

As shown by previous works (Gao et al., 2024; Templeton et al., 2024a), a single SAE activation can capture one monosemantic feature. In this work, we hypothesise a combination of a set of SAE activations can be responsible for a functional feature, such as knowledge selection in case of conflict. Our hypothesis is motivated by Task Vector (Hendel et al., 2023; Todd et al., 2024), which shows that hidden states contain the functional information that drives a task.

We now show how we find the SAE activations that are responsible for driving the knowledge se-

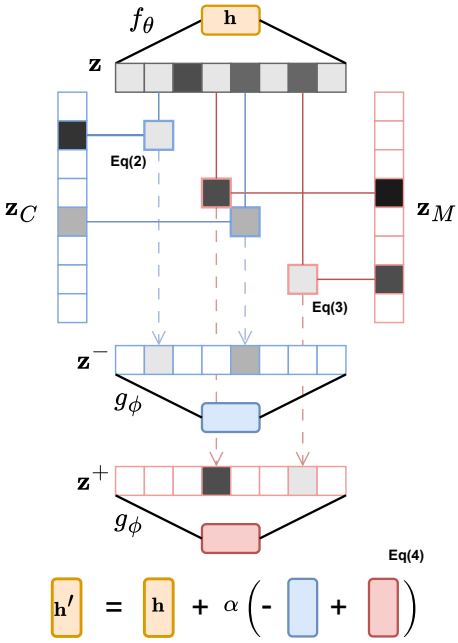


Figure 3: The workflow of SPARE steers the knowledge selection behaviour. The figure presents an example of steering the model to use parametric knowledge. First, the SAE encoder f_θ encodes hidden state $\mathbf{h}$ into the SAE activation $\mathbf{z}$. Then, it determines the values of SAE activations $\mathbf{z}^-$ and $\mathbf{z}^+$ for editing (Eq. (2) and Eq. (3)). Finally, we edit the hidden state using the features extracted from the SAE decoder g_ϕ (Eq. (4)).

lection. First, we calculate mutual information between each SAE activation and the knowledge selection behaviours, which measures to which extent the behaviour depends on each activation. Let the random variable Z_i be the ith activation of SAE, and $Y = \{C, M\}$ be the generated answers; we calculate the mutual information $I(Z_i; Y)$ between them. A higher $I(Z_i; Y)$ indicates a higher dependency between Z_i and the knowledge selection behaviour. We then select the top-k activations with the highest $I(Z_i; Y)$, denoted as $\mathcal{Z}$. More details are available in Appendix C.2

In the following, we determine which knowledge selection behaviour each $Z_i \in \mathcal{Z}$ positively correlates with. Given the sets of activations $\{\mathbf{z}_C^j\}_{j=1}^N$ and $\{\mathbf{z}_M^j\}_{j=1}^N$, we estimate the expected value of each activation feature $Z_i \in \mathcal{Z}$ in both sets, denoted as $\mathbb{E}_C[Z_i]$ and $\mathbb{E}_M[Z_i]$. We then have that Z_i is positively correlated with the behaviour of selecting contextual knowledge if $\mathbb{E}_C[Z_i] - \mathbb{E}_M[Z_i] > 0$. Conversely, if this condition is not met, Z_i is positively correlated with the behaviour of selecting parametric knowledge. Finally, we construct two functional SAE activations $\mathbf{z}_C$ and $\mathbf{z}_M \in \mathbb{R}^n$, that steer the usage of contextual and parametric knowledge, respectively. For each element, z_{Ci} and z_{Mi} are set to 0 if $Z_i \notin \mathcal{Z}$, and the remaining values are taken from $\overline{\mathbf{z}_C}$ and $\overline{\mathbf{z}_M}$ based on their expectations:

$$z_{Ci} = \begin{cases} \overline{z_C}_i, & \text{if } \mathbb{E}_C[Z_i] - \mathbb{E}_M[Z_i] > 0 \\ 0, & \text{otherwise} \end{cases}$$

$$z_{Mi} = \begin{cases} \overline{z_M}_i, & \text{if } \mathbb{E}_C[Z_i] - \mathbb{E}_M[Z_i] < 0 \\ 0, & \text{otherwise} \end{cases}$$

4.3 Editing Activations to Steer Behaviours

In the following, we introduce how we utilise the functional activation $\mathbf{z}_C$ and $\mathbf{z}_M$ to control the usage of knowledge sources at inference time. Suppose we want to control the LLM to use its parametric knowledge and ignore the conflict contextual knowledge that might be misinformation. In this case, we aim to *remove* the features that steer the contextual knowledge usage and *add* the features that steer the parametric knowledge usage. To avoid removing or adding unnecessary features, we restrict the values to edit by the following two constraints. Let an activation be $\mathbf{h}$ and corresponding SAE activations be $\mathbf{z} = f_\theta(\mathbf{h})$. First, in Eq. (2), we determine the value we need to remove from z_i to avoid the undesired behaviour, i.e., generating contextual knowledge in this case. Here, we confirm a non-negative activation after removing, i.e., subtract at most z_i when $z_i < z_{Ci}$:

$$z_i^- = \min\{z_i, z_{Ci}\}. \tag{2}$$

Then, in Eq. (3), we determine the value we need to add to z_i to encourage the desired behaviour, i.e., generating parametric knowledge in this case. Here, we do not add the exceeded value when the activation has reached the value of z_{Mi}:

$$z_i^+ = \max\{z_{Mi} - z_i, 0\}. \tag{3}$$

Finally, we obtain the edited hidden states $\mathbf{h}'$ by:

$$\mathbf{h}' = \mathbf{h} + \alpha\left(-g_\phi\left(\mathbf{z}^-\right) + g_\phi\left(\mathbf{z}^+\right)\right), \tag{4}$$

where $\alpha \in \mathbb{R}^+$ is a user-defined hyperparameter that controls the degree of editing. Note that we do not directly edit the activation $\mathbf{z}$ of hidden state $\mathbf{h}$ to obtain the modified hidden states by $\mathbf{h}' = g_\phi(\mathbf{z}')$ with $\mathbf{z}' = \mathbf{z} - \mathbf{z}^- + \mathbf{z}^+$ for two reasons: *1)* it can result in unexpected information loss of original $\mathbf{h}$ due to the reconstruction loss of the SAE, leading to a nearly zero accuracy in our experiments; *2)* the

definition of the SAE activation shown in Eq. (1) requires each element of $\mathbf{z}$ be a non-negative value, which prevents us from using α to flexibly control the strengthen of editing like Eq. (4).

Similarly, if we want to control the model to be faithful to the context in the case of the contextual knowledge is more likely correct, we can swap z_{Ci} and z_{Mi} in Eq. (2) and Eq. (3).

5 Experimental Results

5.1 Settings

Datasets We use two widely adopted open-domain question-answering datasets with knowledge conflicts NQSwap (Longpre et al., 2021) and Macnoise (Hong et al., 2024) to investigate the controlling capability of several methods.

Models We evaluate our method using Llama3-8B (Dubey et al., 2024) and Gemma2-9B (Rivière et al., 2024), which have corresponding public pre-trained SAEs. Moreover, we also evaluate our method using Llama2-7B (Touvron et al., 2023) with our pre-trained SAEs to examine the feasibility of adopting SPARE to an LLM without public SAEs. More details are presented in Appendix B.

Evaluation We use the greedy decoding method to evaluate the LLMs in an open-ended generation setting. We use 3 in-context demonstrations to align the answer format. The demonstrations use non-conflict evidence E_M and memorised answer M, so they do not point out which knowledge source to select. More details are presented in Appendix C.4. The test examples use E_C, leading to a knowledge conflict for LLMs, and a behaviour-controlling method needs to steer the usage of either parametric or contextual knowledge to generate the answer. We compare the evaluation results under control with the results without any control to show each method's controlling capability.

Baselines We compare SPARE against the following inference-time *representation engineering* methods: 1) TaskVec (Hendel et al., 2023); 2) ActAdd (Turner et al., 2023a); 3) SEA (Qiu et al., 2024) with linear and non-linear versions, noted by subscript "linear" and "SqExp". We compare with the following *contrastive decoding* methods: 1) DoLa (Chuang et al., 2024); 2) CAD (Shi et al., 2024). Moreover, we also compare using in-context learning (ICL) (Brown, 2020) to steer the knowledge selection. We use E_C and C in the demonstrations to guide the model to ignore its parametric knowledge and use the contextual knowledge, and use E_C and M to guide the model to ignore the contextual knowledge and use its parametric knowledge. ICL is not an inference-time strategy because it requires changing the original input of the model to achieve a desired behaviour. More details of baseline implementation and hyperparameters searching are presented in Appendix C.6.

Hyperparameters We select the proper hyperparameters for SPARE in the developments set that is also used to select the hyperparameters of baselines, and the details are presented in Appendix C.6. In the following, we apply SPARE from the 13th to the 16th layer for Llama2-7B and Llama3-8B and from the 23rd to 25th and the 29th to 31st layers for Gemma2-9B; we analyse the performance of editing individual layers in Section 5.4.

5.2 Overall Performance Comparison

Metrics We use Exact Match (EM) to evaluate the performance. Specifically, we evaluate the control capability of generating contextual or parametric answers using the following metrics:

EM$_C$ accuracy of steering the usage of contextual knowledge to generate answers C.

EM$_M$ accuracy of steering the usage of parametric knowledge to generate answer M.

Experimental Results We present the main results in Table 1. First, we find SPARE *outperforms existing representation engineering methods* TaskVec, ActAdd and SEA on steering the usage of both contextual and parametric knowledge. This indicates that SPARE can more accurately isolate features through the SAE and edit specific monosemantic activations to steer the behaviours of LLMs.

Second, we find SPARE *outperforms contrastive decoding methods* DoLa and CAD, especially in steering the usage of parametric knowledge. Though contrastive decoding strategies can effectively improve the use of contextual knowledge, they struggle to steer the parametric knowledge usage. For Gemma2-9B, DoLa even leads to significant degradation, indicating that comparing the logits of intermediate layers for decoding is unstable. SPARE can more effectively steer the usage of both knowledge by adding and removing the desired and undesired functional features, which we further analyse in the later ablation study.

Moreover, SPARE *surpasses the non-inference-time controlling method* ICL. It suggests the

Metric	Method	NQSwap (Longpre et al., 2021)			Macnoise (Hong et al., 2024)		
		Llama3-8B	Llama2-7B	Gemma-2-9B	Llama3-8B	Llama2-7B	Gemma-2-9B
Steer to Use Parametric Knowledge							
	Without Controlling	$26.63_{\pm 6.02}$	$22.23_{\pm 4.75}$	$26.32_{\pm 1.80}$	$18.96_{\pm 2.65}$	$22.37_{\pm 1.89}$	$17.06_{\pm 3.79}$
EM_M	TaskVec (Hendel et al., 2023)	$24.16_{\pm 6.58}$	$24.88_{\pm 0.85}$	$29.85_{\pm 0.83}$	$21.23_{\pm 1.89}$	$22.93_{\pm 2.31}$	$28.92_{\pm 1.19}$
	ActAdd (Turner et al., 2023a)	$37.87_{\pm 8.96}$	$31.43_{\pm 3.68}$	$27.67_{\pm 0.82}$	$26.17_{\pm 0.22}$	$27.52_{\pm 3.07}$	$29.75_{\pm 1.68}$
	SEA_{linear} (Qiu et al., 2024)	$21.03_{\pm 1.83}$	$23.73_{\pm 0.86}$	$24.43_{\pm 0.91}$	$12.84_{\pm 0.18}$	$15.64_{\pm 0.24}$	$28.10_{\pm 2.78}$
	SEA_{SqExp} (Qiu et al., 2024)	$13.64_{\pm 1.62}$	$16.66_{\pm 0.55}$	$23.79_{\pm 1.38}$	$14.24_{\pm 1.45}$	$16.24_{\pm 1.06}$	$28.07_{\pm 1.30}$
	DoLa (Chuang et al., 2024)	$25.53_{\pm 5.19}$	$16.50_{\pm 3.91}$	$20.58_{\pm 1.06}$	$16.52_{\pm 2.65}$	$15.66_{\pm 0.88}$	$19.81_{\pm 2.58}$
	$^{\flat}$CAD (Shi et al., 2024)	$33.72_{\pm 0.84}$	$31.23_{\pm 1.45}$	$41.17_{\pm 0.59}$	$28.58_{\pm 0.75}$	$30.81_{\pm 0.94}$	$\underline{33.15}_{\pm 2.87}$
	$^{\sharp}$ICL (Brown, 2020)	$\underline{43.73}_{\pm 1.55}$	$\underline{31.67}_{\pm 5.49}$	$\underline{43.10}_{\pm 3.63}$	$\underline{29.54}_{\pm 4.16}$	$\underline{31.23}_{\pm 0.94}$	$21.91_{\pm 2.35}$
	SPARE (Ours)	$\mathbf{47.51}_{\pm 1.30}$	$\mathbf{43.76}_{\pm 3.14}$	$\mathbf{44.11}_{\pm 1.30}$	$\mathbf{30.72}_{\pm 1.42}$	$\mathbf{35.43}_{\pm 1.10}$	$\mathbf{35.53}_{\pm 2.07}$
Steer to Use Contextual Knowledge							
	Without Controlling	$42.69_{\pm 8.40}$	$41.67_{\pm 4.66}$	$45.96_{\pm 2.48}$	$69.36_{\pm 3.57}$	$62.38_{\pm 3.05}$	$59.25_{\pm 2.82}$
EM_C	TaskVec (Hendel et al., 2023)	$41.88_{\pm 9.45}$	$38.25_{\pm 1.23}$	$45.52_{\pm 1.06}$	$\underline{88.47}_{\pm 1.93}$	$\underline{86.91}_{\pm 0.44}$	$59.25_{\pm 1.49}$
	ActAdd (Turner et al., 2023a)	$51.91_{\pm 8.03}$	$47.48_{\pm 3.93}$	$46.90_{\pm 1.89}$	$73.01_{\pm 1.58}$	$69.64_{\pm 0.20}$	$59.66_{\pm 2.89}$
	SEA_{linear} (Qiu et al., 2024)	$43.61_{\pm 10.3}$	$47.73_{\pm 0.43}$	$52.95_{\pm 1.90}$	$69.78_{\pm 0.97}$	$67.32_{\pm 0.28}$	$60.31_{\pm 2.25}$
	SEA_{SqExp} (Qiu et al., 2024)	$57.08_{\pm 2.92}$	$48.04_{\pm 0.45}$	$61.45_{\pm 0.54}$	$72.04_{\pm 1.60}$	$68.20_{\pm 1.10}$	$61.45_{\pm 0.30}$
	DoLa (Chuang et al., 2024)	$44.29_{\pm 8.46}$	$33.54_{\pm 3.38}$	$15.90_{\pm 10.1}$	$68.45_{\pm 3.83}$	$50.95_{\pm 5.15}$	$23.34_{\pm 10.5}$
	$^{\flat}$CAD (Shi et al., 2024)	$65.65_{\pm 5.50}$	$54.69_{\pm 3.25}$	$63.10_{\pm 2.32}$	$78.69_{\pm 3.85}$	$70.07_{\pm 3.77}$	$\underline{64.12}_{\pm 4.44}$
	$^{\sharp}$ICL (Brown, 2020)	$\underline{73.35}_{\pm 3.82}$	$\underline{63.33}_{\pm 3.50}$	$\underline{70.19}_{\pm 2.51}$	$51.75_{\pm 5.60}$	$47.51_{\pm 1.86}$	$47.24_{\pm 3.81}$
	SPARE (Ours)	$\mathbf{77.69}_{\pm 1.24}$	$\mathbf{69.32}_{\pm 1.26}$	$\mathbf{73.78}_{\pm 0.74}$	$\mathbf{92.24}_{\pm 0.49}$	$\mathbf{87.30}_{\pm 1.96}$	$\mathbf{87.96}_{\pm 1.85}$

Table 1: Overall performance of steering the utilisation of parametric and contextual knowledge, measured by EM_M and EM_C. "Without Controlling" indicates the baseline that we do not use any controlling methods to steer the generation. $^{\sharp}$ICL is not an inference-time controlling strategy, which controls the behaviours by changing demonstrations. $^{\flat}$CAD needs additional forwarding for contrastive decoding.

SPARE can both effectively and efficiently control the knowledge selection behaviours of LLMs. It also suggests a promising capability of representation engineering to control the behaviours of LLMs at inference time in practical applications.

5.3 Multi-Perspective Controlling Analysis

In the following, we analyse the controlling capability of SPARE from different perspectives: *1)* the capability of changing the behaviour (Fig. 4a), *2)* the potential negative impact of intervention (Fig. 4b), and *3)* the ablation study (Fig. 4c).

Capability of Changing the Behaviours Unlike merely comparing overall performance across the entire dataset, we further examine their capability of changing the original knowledge selection behaviour of LLMs by the following two metrics:

$EM_{C\rightarrow M}$ accuracy of changing the behaviour from generating contextual answers C to parametric answers M in the subset of instances where the model generates C without controlling.

$EM_{M\rightarrow C}$ accuracy of changing the behaviour from generating parametric answers M to contextual answers C in the subset of instances where the model generates M without controlling.

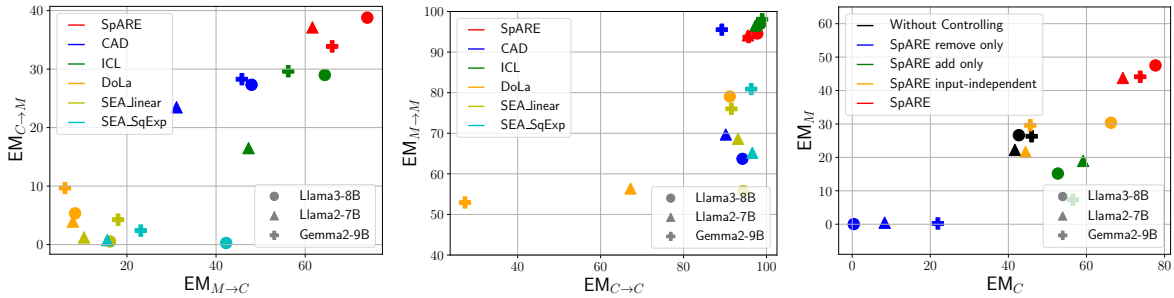
As shown in Fig. 4a, SPARE is located in the upper-right area of the figure, outperforming contrastive decoding and other representation engineering methods. It suggests that SPARE can more effectively change both the original behaviours of using contextual and parametric knowledge. We also observe that all methods are less effective in steering toward the use of parametric knowledge than contextual knowledge. This finding matches the previous works (Su et al., 2024; Xie et al., 2024a; Ortu et al., 2024), which shows LLMs prefer contextual knowledge, and thus more difficult to steer the behaviour of using parametric knowledge.

Impacts of Intervention A sufficient but unnecessary intervention can change the behaviour of LLMs, but it also can introduce noise and decrease accuracy. Here, we investigate the potential negative impact of methods by steering LLMs using the same knowledge they will use without control. We expect LLMs to maintain their original behaviours, measured by the following two metrics:

$EM_{M\rightarrow M}$ accuracy of maintaining the behaviour of generating M when steering the use of parametric knowledge in the subset of instances where the model generates M without controlling.

$EM_{C\rightarrow C}$ accuracy of maintaining the behaviour of generating C when steering the use contextual knowledge in the subset of instances where the model generates C without controlling.

As shown in Fig. 4b, ICL is the most stable method that nearly does not change the original be-



(a) Capability of changing behaviours. (b) Impact of intervention. (c) Ablation studies.

Figure 4: Detailed evaluation results of controlling capability on NQSwap. We use different colours for different methods and use different shapes for different models. The upper-right area indicates a high performance for all figures. (a) presents the capability of changing the behaviour of LLMs, where x-axis and y-axis are $\text{EM}_{C\to M}$ and $\text{EM}_{M\to C}$, measuring the capability of changing the answer from C to M and from M to C, respectively; (b) presents the capability of maintaining the behaviour when steering to the same behaviour as the original behaviour, where x-axis and y-axis are $\text{EM}_{M\to M}$ and $\text{EM}_{C\to C}$, measuring the maintaining capability of generating M and C, respectively; (c) present the ablation analysis of SPARE, x-axis and y-axis are EM_M and EM_C.

haviour when steering the same behaviour. SPARE has a close performance to ICL, indicating it can steer the behaviour effectively while introducing a little unnecessary editing. Though CAD maintains the most accuracy in contextual knowledge, its performance decreases substantially in maintaining the behaviour of generating parametric knowledge. Finally, while other representation engineering methods may alter the entire model behaviour due to editing of polysemantic features, SPARE provides a more precise approach to editing through the SAE activations and thus delivers better performance in maintaining the model behaviour.

Ablation Study We present the ablation study on the following settings: *1)* SPARE input-independent: it uses $\mathbf{z}_C$ and $\mathbf{z}_M$ to steer the generation without calculating $\mathbf{z}^-$ and $\mathbf{z}^+$ based on the input activation; *2)* SPARE remove only: it edits the hidden states by only removing the functional features of the undesired behaviour; *3)* SPARE add only: it edits the hidden states by only adding the functional features of the desired behaviour.

As shown in Fig. 4c, we can see that every ablation results in a significant controlling capability decrease. The input-independent editing strategy that omits the calculations of Eq. (2) and Eq. (3) fails to steer the usage of knowledge and obtain results that are close to without controlling. The results of SPARE "removed only" obtain a zero accuracy on both EM_M and EM_C, which indicates the model cannot keep the original behaviour and also cannot generate answers toward another behaviour. It suggests the SPARE can effectively remove the functional features of the original behaviour. Moreover, SPARE "add only" leads to worse performance than without controlling, suggesting the importance of removing the features of undesired knowledge selection behaviour.

5.4 Analysing the Layer Choice

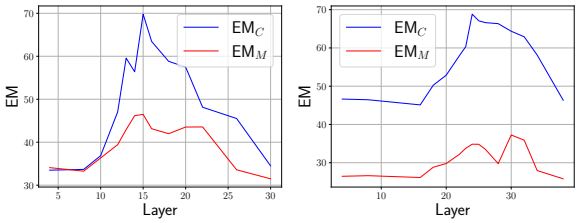
We present the results of editing multiple layers in Table 1; here, we analyse the effectiveness of SPARE by editing each layer individually. As shown in Fig. 5, we find SPARE can control the behaviour of LLMs most effectively at mid-layers for both Llama3-8B and Gemma2-9B. These layers are also where we can detect knowledge conflict most accurately, as shown in Fig. 2 and Appendix A. This supports the practical application of inference-time knowledge selection behaviour controlling, where SPARE can effectively steer the generation once we detect the conflict.

The effectiveness of steering the behaviours in middle layers also matches previous findings (Hendel et al., 2023; Pan et al., 2023; Todd et al., 2024) suggesting that the middle layers of LLMs contain the functional feature that drives a task. To the best of our knowledge, we are the first to accurately extract this functional feature using SAE.

5.5 Analysing the Residual Stream

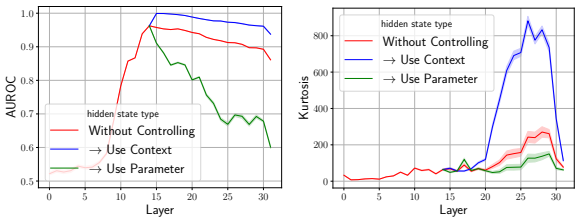
We analyse how the residual stream changes after applying SPARE. Here, we edit the hidden states from $\mathcal{D}_{E_C} = \{(Q, E_C)\}$ at the 15th layer to steer the contextual and parametric knowledge usage.

In Fig. 6a, we present the probing results on the residual stream using the same probing model de-



(a) Llama3-8B (b) Gemma2-9B

Figure 5: Effectiveness of SPARE on editing different layers individually.



(a) Probing results (b) Skewness pattern

Figure 6: The residual stream changes after applying SPARE at the 15th layer.

scribed in Section 3. We observe that when we steer the usage of parametric knowledge, the probing performance decreases (green line), indicating the residual stream contains less conflict knowledge, and making the probing model unable to distinguish whether a given activation is from $\mathcal{D}_{E_C}$ or $\mathcal{D}_{E_M}$. In contrast, when we steer the usage of contextual knowledge, the probing performance increases (blue line), indicating that the representations of activations become more different from $\mathcal{D}_{E_M}$, making the probing model easier to distinguish between them.

In Fig. 6b, we find the skewness of the representation from the residual stream shows distinct patterns after applying SPARE. When we apply SPARE to steer the usage of contextual knowledge at the 15th layer, the residual stream becomes significantly more skewed starting from the 19th layer (blue line); in contrast, when we use parametric knowledge, the residual stream becomes less skewed (red line). This finding indicates that while SPARE changes the representation characteristics of the residual stream, the results appear later than the editing. Here, we only provide our empirical observation on the representation pattern and leave investigating the reasons in future works. We provide more analysis in Appendix F.

6 Related Works

Representation Engineering Several works show that neural networks can acquire emergent, semantically meaningful internal structures (Mikolov et al., 2013; Bolukbasi et al., 2016; Schramowski et al., 2019; Radford et al., 2017; Dai et al., 2022). Some works modify the activation of a trained model, altering its behaviour (Ravfogel et al., 2020; Iskander et al., 2023; Liu et al., 2024; Zou et al., 2023b; Li et al., 2023), and some works extract latent vectors directly from the trained model and leveraging these vectors to regulate the model's inference (Turner et al., 2023b; Subramani et al., 2022; Rimsky et al., 2023).

Knowledge Conflicts Knowledge conflicts refer to discrepancies among contextual and parametric knowledge (Chen et al., 2022; Xie et al., 2024b; Su et al., 2024). Xu et al. (2024) identifies three types of knowledge conflicts: *inter-context*, *context-memory*, and *intra-memory* conflicts; in this work, we focus on context-memory knowledge conflicts.

Sparse Auto-Encoders SAEs have been introduced as a post-hoc analysis tool to identify disentangled features within uncompressed representations of an LLM (Yun et al., 2021; Bricken et al., 2023; Huben et al., 2024). SAEs are trained with sparsity regularisation to learn a sparse, overcomplete basis that characterises the activation space of an LLM (Bereska and Gavves, 2024). Marks et al. (2024) showed that the features learned by SAEs can identify sparse circuits in LLMs. Similarly, Templeton et al. (2024b) uses monosemantic features and steering LLMs' generation accordingly.

7 Conclusions

We investigated the context-memory knowledge conflicts in LLMs. We identify that knowledge conflicts can be detected by probing the residual stream of the model (Section 3) and propose SPARE (Section 4), a training-free representation engineering method that leverages pre-trained SAEs to control the knowledge selection behaviour of LLMs. Our experimental results on ODQA tasks show that SPARE produces more accurate results than existing representation engineering and contrastive decoding methods. By providing a mechanism to steer knowledge selection behaviours at inference time, SPARE offers a promising approach to managing knowledge conflicts in LLMs without significant computational overhead.

Limitations

While our proposed method, SPARE, demonstrates effective control over knowledge selection behaviours in LLMs, there are several limitations to consider. First, the approach relies on pre-trained SAEs to identify and manipulate functional features within the internal activations of the model. Training SAEs can be computationally intensive, especially for large models, and may not be practical in all settings. Additionally, the requirement of pre-trained SAEs means that the method may not directly apply to models where pre-trained SAEs are unavailable or cannot be efficiently trained. Second, our experiments are conducted on specific ODQA tasks involving context-memory knowledge conflicts. While the results are promising in this setting, it is unclear how well the method generalizes to other types of tasks or conflicts, such as those involving complex reasoning, multi-hop questions, or different modalities. Further research is needed to assess the applicability of SPARE in broader contexts. Finally, the control over knowledge selection behaviours is evaluated primarily in terms of steering the model towards either contextual or parametric knowledge. In real-world applications, the decision about which knowledge source to trust may not be binary and may require a more elaborate approach, such as using another model as a critic (Hong et al., 2024).

References

Zeyuan Allen-Zhu and Yuanzhi Li. 2023. Physics of language models: Part 1, context-free grammar. *arXiv preprint arXiv:2305.13673*.

Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Leonard Bereska and Efstratios Gavves. 2024. Mechanistic interpretability for AI safety - A review. *CoRR*, abs/2404.14082.

Tolga Bolukbasi, Kai-Wei Chang, James Y. Zou, Venkatesh Saligrama, and Adam Tauman Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *NIPS*, pages 4349–4357.

Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nicholas L. Turner, Cem Anil, Carson Denison, Amanda Askell, Robert Lasenby, Yifan Wu, Shauna Kravec, Nicholas Schiefer, Tim Maxwell, Nicholas Joseph, Alex Tamkin, Karina Nguyen, Brayden McLean, Josiah E. Burke, Tristan Hume, Shan Carter, Tom Henighan, and Chris Olah. 2023. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*.

Tom B Brown. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Canyu Chen and Kai Shu. 2023a. Can llm-generated misinformation be detected? *arXiv preprint arXiv:2309.13788*.

Canyu Chen and Kai Shu. 2023b. Combating misinformation in the age of llms: Opportunities and challenges. *AI Magazine*.

Hung-Ting Chen, Michael J. Q. Zhang, and Eunsol Choi. 2022. Rich knowledge sources bring complex knowledge conflicts: Recalibrating models to reflect conflicting evidence. In *EMNLP*, pages 2292–2307. Association for Computational Linguistics.

Yung-Sung Chuang, Yujia Xie, Hongyin Luo, Yoon Kim, James R. Glass, and Pengcheng He. 2024. Dola: Decoding by contrasting layers improves factuality in large language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single vector: Probing sentence embeddings for linguistic properties. *arXiv preprint arXiv:1805.01070*.

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. 2022. Knowledge neurons in pretrained transformers. In *ACL (1)*, pages 8493–8502. Association for Computational Linguistics.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, et al. 2021. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 1(1):12.

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

Roee Hendel, Mor Geva, and Amir Globerson. 2023. In-context learning creates task vectors. In *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pages 9318–9333. Association for Computational Linguistics.

Giwon Hong, Jeonghwan Kim, Junmo Kang, Sung-Hyon Myaeng, and Joyce Jiyoung Whang. 2024. Why so gullible? enhancing the robustness of retrieval-augmented models against counterfactual noise. In *Findings of the Association for Computational Linguistics: NAACL 2024, Mexico City, Mexico, June 16-21, 2024*, pages 2474–2495. Association for Computational Linguistics.

Robert Huben, Hoagy Cunningham, Logan Riggs, Aidan Ewart, and Lee Sharkey. 2024. Sparse autoencoders find highly interpretable features in language models. In *ICLR*. OpenReview.net.

Shadi Iskander, Kira Radinsky, and Yonatan Belinkov. 2023. Shielded representations: Protecting sensitive attributes through iterative gradient-based projection. In *ACL (Findings)*, pages 5961–5977. Association for Computational Linguistics.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick S. H. Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 6769–6781. Association for Computational Linguistics.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Kenneth Li, Oam Patel, Fernanda B. Viégas, Hanspeter Pfister, and Martin Wattenberg. 2023. Inference-time intervention: Eliciting truthful answers from a language model. In *NeurIPS*.

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

Sheng Liu, Haotian Ye, Lei Xing, and James Y. Zou. 2024. In-context vectors: Making in context learning more effective and controllable through latent space steering. In *ICML*. OpenReview.net.

Shayne Longpre, Kartik Perisetla, Anthony Chen, Nikhil Ramesh, Chris DuBois, and Sameer Singh. 2021. Entity-based knowledge conflicts in question answering. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7052–7063, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 9802–9822. Association for Computational Linguistics.

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

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in GPT. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Tomás Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013. Linguistic regularities in continuous space word representations. In *HLT-NAACL*, pages 746–751. The Association for Computational Linguistics.

Chris Olah. 2023. Distributed representations: Composition & superposition. *Transformer Circuits Thread*.

Francesco Ortu, Zhijing Jin, Diego Doimo, Mrinmaya Sachan, Alberto Cazzaniga, and Bernhard Schölkopf. 2024. Competition of mechanisms: Tracing how language models handle facts and counterfactuals. *arXiv preprint arXiv:2402.11655*.

Jane Pan, Tianyu Gao, Howard Chen, and Danqi Chen. 2023. What in-context learning "learns" in-context: Disentangling task recognition and task learning. In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 8298–8319. Association for Computational Linguistics.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick S. H. Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander H. Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 2463–2473. Association for Computational Linguistics.

Yifu Qiu, Zheng Zhao, Yftah Ziser, Anna Korhonen, Edoardo M. Ponti, and Shay B. Cohen. 2024. Spectral editing of activations for large language model alignment. *CoRR*, abs/2405.09719.

Alec Radford, Rafal Józefowicz, and Ilya Sutskever. 2017. Learning to generate reviews and discovering sentiment. *CoRR*, abs/1704.01444.

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

Shauli Ravfogel, Yanai Elazar, Hila Gonen, Michael Twiton, and Yoav Goldberg. 2020. Null it out: Guarding protected attributes by iterative nullspace projection. In *ACL*, pages 7237–7256. Association for Computational Linguistics.

Nina Rimsky, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Matt Turner. 2023. Steering llama 2 via contrastive activation addition. *CoRR*, abs/2312.06681.

Morgane Rivière, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, Johan Ferret, Peter Liu, Pouya Tafti, Abe Friesen, Michelle Casbon, Sabela Ramos, Ravin Kumar, Charline Le Lan, Sammy Jerome, Anton Tsitsulin, Nino Vieillard, Piotr Stanczyk, Sertan Girgin, Nikola Momchev, Matt Hoffman, Shantanu Thakoor, Jean-Bastien Grill, Behnam Neyshabur, Olivier Bachem, Alanna Walton, Aliaksei Severyn, Alicia Parrish, Aliya Ahmad, Allen Hutchison, Alvin Abdagic, Amanda Carl, Amy Shen, Andy Brock, Andy Coenen, Anthony Laforge, Antonia Paterson, Ben Bastian, Bilal Piot, Bo Wu, Brandon Royal, Charlie Chen, Chintu Kumar, Chris Perry, Chris Welty, Christopher A. Choquette-Choo, Danila Sinopalnikov, David Weinberger, Dimple Vijaykumar, Dominika Rogozinska, Dustin Herbison, Elisa Bandy, Emma Wang, Eric Noland, Erica Moreira, Evan Senter, Evgenii Eltyshev, Francesco Visin, Gabriel Rasskin, Gary Wei, Glenn Cameron, Gus Martins, Hadi Hashemi, Hanna Klimczak-Plucinska, Harleen Batra, Harsh Dhand, Ivan Nardini, Jacinda Mein, Jack Zhou, James Svensson, Jeff Stanway, Jetha Chan, Jin Peng Zhou, Joana Carrasqueira, Joana Iljazi, Jocelyn Becker, Joe Fernandez, Joost van Amersfoort, Josh Gordon, Josh Lipschultz, Josh Newlan, Ju-yeong Ji, Kareem Mohamed, Kartikeya Badola, Kat Black, Katie Millican, Keelin McDonell, Kelvin Nguyen, Kiranbir Sodhia, Kish Greene, Lars Lowe Sjösund, Lauren Usui, Laurent Sifre, Lena Heuermann, Leticia Lago, and Lilly McNealus. 2024. Gemma 2: Improving open language models at a practical size. *CoRR*, abs/2408.00118.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2024. Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems*, 36.

Patrick Schramowski, Cigdem Turan, Sophie F. Jentzsch, Constantin A. Rothkopf, and Kristian Kersting. 2019. BERT has a moral compass: Improvements of ethical and moral values of machines. *CoRR*, abs/1912.05238.

Weijia Shi, Xiaochuang Han, Mike Lewis, Yulia Tsvetkov, Luke Zettlemoyer, and Wen-tau Yih. 2024. Trusting your evidence: Hallucinate less with context-aware decoding. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Short Papers, NAACL 2024, Mexico City, Mexico, June 16-21, 2024*, pages 783–791. Association for Computational Linguistics.

Zhaochen Su, Jun Zhang, Xiaoye Qu, Tong Zhu, Yanshu Li, Jiashuo Sun, Juntao Li, Min Zhang, and Yu Cheng. 2024. Conflictbank: A benchmark for evaluating the influence of knowledge conflicts in llm. *arXiv preprint arXiv:2408.12076*.

Nishant Subramani, Nivedita Suresh, and Matthew E. Peters. 2022. Extracting latent steering vectors from pretrained language models. In *ACL (Findings)*, pages 566–581. Association for Computational Linguistics.

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

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

Eric Todd, Millicent L. Li, Arnab Sen Sharma, Aaron Mueller, Byron C. Wallace, and David Bau. 2024. Function vectors in large language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Alexander Matt Turner, Lisa Thiergart, David Udell, Gavin Leech, Ulisse Mini, and Monte MacDiarmid. 2023a. Activation addition: Steering language models without optimization. *CoRR*, abs/2308.10248.

Alexander Matt Turner, Lisa Thiergart, David Udell, Gavin Leech, Ulisse Mini, and Monte MacDiarmid.

2023b. Activation addition: Steering language models without optimization. *CoRR*, abs/2308.10248.

Yike Wang, Shangbin Feng, Heng Wang, Weijia Shi, Vidhisha Balachandran, Tianxing He, and Yulia Tsvetkov. 2023. Resolving knowledge conflicts in large language models. *CoRR*, abs/2310.00935.

Yuxiang Wu, Yu Zhao, Baotian Hu, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2022. An efficient memory-augmented transformer for knowledge-intensive NLP tasks. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 5184–5196. Association for Computational Linguistics.

Jian Xie, Kai Zhang, Jiangjie Chen, Renze Lou, and Yu Su. 2024a. Adaptive chameleon or stubborn sloth: Revealing the behavior of large language models in knowledge conflicts. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Jian Xie, Kai Zhang, Jiangjie Chen, Renze Lou, and Yu Su. 2024b. Adaptive chameleon or stubborn sloth: Revealing the behavior of large language models in knowledge conflicts. In *ICLR*. OpenReview.net.

Rongwu Xu, Zehan Qi, Cunxiang Wang, Hongru Wang, Yue Zhang, and Wei Xu. 2024. Knowledge conflicts for llms: A survey. *arXiv preprint arXiv:2403.08319*.

Zeyu Yun, Yubei Chen, Bruno Olshausen, and Yann LeCun. 2021. Transformer visualization via dictionary learning: contextualized embedding as a linear superposition of transformer factors. In *Proceedings of Deep Learning Inside Out (DeeLIO): The 2nd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 1–10, Online. Association for Computational Linguistics.

Wanru Zhao, Vidit Khazanchi, Haodi Xing, Xuanli He, Qiongkai Xu, and Nicholas Donald Lane. 2024. Attacks on third-party apis of large language models. In *ICLR 2024 Workshop on Secure and Trustworthy Large Language Models*.

Zexuan Zhong, Ziqing Huang, Alexander Wettig, and Danqi Chen. 2023. Poisoning retrieval corpora by injecting adversarial passages. *arXiv preprint arXiv:2310.19156*.

Zeyuan Allen Zhu and Yuanzhi Li. 2023. Physics of language models: Part 3.1, knowledge storage and extraction. *arXiv preprint arXiv:2309.14316*.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, Shashwat Goel, Nathaniel Li, Michael J. Byun, Zifan Wang, Alex Mallen, Steven Basart, Sanmi Koyejo, Dawn Song, Matt Fredrikson, J. Zico Kolter, and Dan Hendrycks. 2023a. Representation engineering: A top-down approach to AI transparency. *CoRR*, abs/2310.01405.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, Shashwat Goel, Nathaniel Li, Michael J. Byun, Zifan Wang, Alex Mallen, Steven Basart, Sanmi Koyejo, Dawn Song, Matt Fredrikson, J. Zico Kolter, and Dan Hendrycks. 2023b. Representation engineering: A top-down approach to AI transparency. *CoRR*, abs/2310.01405.

Wei Zou, Runpeng Geng, Binghui Wang, and Jinyuan Jia. 2024. Poisonedrag: Knowledge poisoning attacks to retrieval-augmented generation of large language models. *arXiv preprint arXiv:2402.07867*.

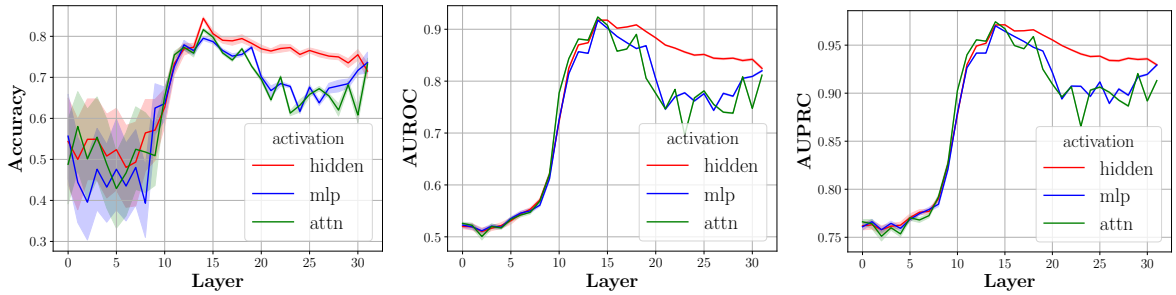


Figure 7: Knowledge conflict probing results using Llama2-7B on NQSwap.

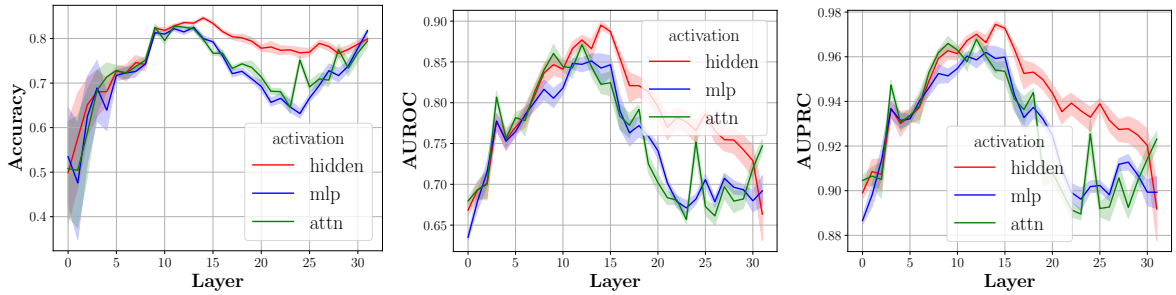


Figure 8: Knowledge conflict probing results using Llama2-7B on Macnoise.

A More Analysis of Knowledge Conflict Probing

A.1 Details of Probing Model

We train the probing model with an L_1 norm regularisation for all probing experiments. The training objective is $\mathcal{L} = -\log P(y = y_i) + \lambda \|W\|_1$, where we set λ to 3×10^{-4} and y_i is the label. We train 20 times with different random seeds for each probing task, and we report the average and deviation in our experiments. We split the training and test datasets for the probing tasks, ensuring no overlapping questions between them.

A.2 More Probing Results

We present the knowledge conflict probing results of Llama2-7B on NQSwap and Macnoise on Fig. 7 and Fig. 8 using accuracy, AUROC and AUPRC as metrics.

B Sparse Auto-Encoders Details

Both the SAEs of Llama3-8B from ElethuerAI and Llama2-7B pre-trained by us have a latent dimension $n = 131072$, 32 times larger than the number of dimensions of the hidden state size $d = 4096$, and use ReLU as activation function σ (Eq. (1)). Gemma2-9B uses JumpReLU (Rajamanoharan et al., 2024) as the activation function. GemmaScope (Lieberum et al., 2024) provides different sizes of SAEs, and we use the size of $n = 131072$ in the experiment.

Following Gao et al. (2024), we pre-train the SAEs for Llama2-7B with TopK activation functions: $\mathbf{z} = \text{TopK}\left(\mathbf{W}_\theta(\mathbf{h} - \mathbf{b})\right)$, which only keeps the k largest latents during pre-training, while does not use sparsity regularisation during pre-training. The loss is calculated by the L_2 norm of reconstruction error: $\mathcal{L} = \|\mathbf{h} - \hat{\mathbf{h}}\|_2^2$. We pre-train SAE models[2] with 10B tokens sampled from RedPajama[3] and use the hyperparamters determined by Gao et al. (2024). The pre-training for an SAE for a certain layer of hidden states costs about 300 80G A100 GPU hours.

Though it costs non-trivial resources for pre-training SAEs, we believe it is valuable to explore using SAEs to resolve knowledge conflicts for the following reasons: *1)* SAEs are general models for interpreting the representation of LLMs, which have broader applications beyond steering knowledge selection behaviours; *2)* SAEs are becoming popular tools for interpreting LLMs with rising numbers of open-resource frameworks and pre-trained models released recently, so we can use the pre-trained SAEs conveniently rather than pre-training SAEs by ourselves in the future.

[2] https://github.com/EleutherAI/sae
[3] https://huggingface.co/datasets/togethercomputer/RedPajama-Data-V2

C Implementation Details

C.1 Collecting Activations

Collecting activations is an essential step in representation engineering methods (Zou et al., 2023a; Qiu et al., 2024), where we will extract desired features from the collected activations and use these features to edit the activations of LLMs. In SPARE, we first sample demonstrations from the development set using 5 seeds. Then, we use these demonstrations to test the rest of the questions with conflict evidence E_C. Based on the predictions, we split the instance into $\mathcal{D}_C$ and $\mathcal{D}_M$ and collect their corresponding hidden states, denoting as $\{\mathbf{h}_C^j\}_{j=1}^N$ and $\{\mathbf{h}_M^j\}_{j=1}^N$. Then, the corresponding layer's SAE encodes them to $\{\mathbf{z}_C^j\}_{j=1}^N$ and $\{\mathbf{z}_M^j\}_{j=1}^N$. Here, we use j to index the collected instances and use i to index the SAE's activation.

One strategy to highlight the values of behaviour-related activation is weighted averaging $\{\mathbf{z}_C^j\}_{j=1}^N$ and $\{\mathbf{z}_M^j\}_{j=1}^N$ by the confidence of generating a specific answer. Specifically, denote the confidence of generating answers C and M conditioned on $\mathbf{h}_C^j$ and $\mathbf{h}_M^j$ as

$$\lambda_C^j = \frac{\log P(C \mid \mathbf{h}_C^j)}{\log P(C \mid \mathbf{h}_C^j) + \log P(M \mid \mathbf{h}_C^j)}$$

$$\mu_M^j = \frac{\log P(M \mid \mathbf{h}_M^j)}{\log P(C \mid \mathbf{h}_M^j) + \log P(M \mid \mathbf{h}_M^j)},$$

where $P(\cdot \mid \mathbf{h})$ is calculated by the output of LLMs. We use the normalised confidence $\lambda_C^{'j}$ and $\mu_M^{'j}$ as weight to average $\{\mathbf{z}_C^j\}_{j=1}^N$ and $\{\mathbf{z}_M^j\}_{j=1}^N$, respectively:

$$\overline{\mathbf{z}_C} = \sum_{j=1}^N \lambda_C^{'j} \mathbf{z}_C^j, \quad \text{and} \quad \overline{\mathbf{z}_M} = \sum_{j=1}^N \mu_M^{'j} \mathbf{z}_M^j.$$

We find this strategy brings a slight improvement compared to directly average $\{\mathbf{z}_C^j\}_{j=1}^N$ and $\{\mathbf{z}_M^j\}_{j=1}^N$.

C.2 Identifying Functional Activations

We identify the activations that steer the usage of contextual and parametric knowledge by calculating the mutual information between activation Z_i and the generated answer $Y = \{C, M\}$:

$$I(Z_i; Y) = \sum_{z_i \in Z_i} \sum_{y \in \{C,M\}} P(z_i, y) \log \frac{P(z_i, y)}{P(z_i)P(y)}.$$

Since a higher $I(Z_i; Y)$[4] indicates a higher dependency between Z_i and the knowledge selection behaviour, we sorted $\{I(Z_i; Y)\}_{i=1}^n$ in descending order, and consider the top k activation work as functional activations. To decide the number of selected activations k, we introduce a hyperparameter K. We then choose k such that:

$$k = \arg\min_k \sum_{i=1}^k \frac{I(Z_i; Y)}{\sum_{j=1}^n I(Z_j; Y)} \geq K, \quad (5)$$

which means selecting the top k SAE activations with the proportion $K\%$ in the sum of all mutual information $\sum_{j=1}^n I(Z_j; Y)$. One potential improvement is normalising mutual information based on entropy, which has shown better properties for comparing the importance of features. We determine the top activations in each layer individually rather than ranking all mutual information across layers.

The proportion K abstracts the exact number of activations to select. We expect the same types of SAEs, e.g., pre-trained by TopK (Gao et al., 2024) or JumpReLU (Rajamanoharan et al., 2024), can share similar values of K for controlling. We present the relation between k and K in Appendix E. We also present the selected Gemma2-9B SAEs activations used by SPARE in Appendix D.

C.3 Impact of the Size of the Collected Activations

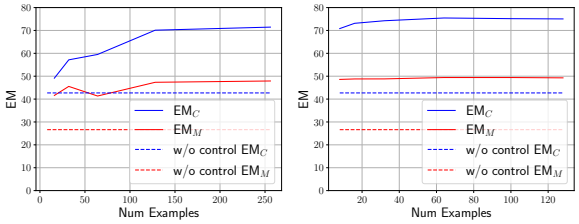
We collect the hidden states $\{\mathbf{h}_C\}_{j=1}^N$ and $\{\mathbf{h}_M\}_{j=1}^N$ to calculate the value of functional activations $\mathbf{z}_C$ and $\mathbf{z}_M$, and we also use $\{\mathbf{z}_C\}_{j=1}^N$ and $\{\mathbf{z}_M\}_{j=1}^N$ to estimate the mutual information. Here, we analyse the impact of N on the controlling performance.

As shown in Fig. 9a, the performance increases when we use 8 examples to 128 for calculating the mutual information, while collecting more activations brings slight improvement. In Fig. 9b, the performance does not increase until using 64 examples to calculate $\mathbf{z}_C$ and $\mathbf{z}_M$. The above analysis indicates that SPARE needs around 128 activations to achieve a high controlling capability.

C.4 Development Set and Demonstrations

We held out a demonstration set consisting of 128 instances from each dataset for each model. Each

[4] We estimate mutual information between continuous variables Z_i and discrete labels Y using https://scikit-learn.org/1.5/modules/generated/sklearn.feature_selection.mutual_info_classif.html



(a) Use different numbers of activations to calculate mutual information.

(b) Use different numbers of activations to calculate $\mathbf{z}_C$ and $\mathbf{z}_M$

Figure 9: The impact of the number of the collected hidden states on the controlling performance.

question in the demonstration set can be answered correctly by the corresponding model without providing evidence; more specifically, we test it with the close-book setting with few-shot examples to align the answer format. We use the non-conflict context E_M and memorised answer M in demonstrations. Since the contextual and parametric knowledge are consistent in the demonstration set, they do not provide information about how to resolve the knowledge conflict for the test examples when presented with conflict evidence E_C. We sample 5 different sets of demonstrations using 5 different seeds and present the average results with deviations. All methods we compared use the same sets of demonstrations for each model.

The held-out demonstration set is also used as the development set to search hyperparameters for each model in each dataset. We sample a set of demonstrations to align the answer format and use the rest of the instances to evaluate the performance.

C.5 Implementation Details of Representation Engineering Baselines

For all representation engineering baselines TaskVec (Hendel et al., 2023), ActAdd (Turner et al., 2023a) and SEA (Qiu et al., 2024), we experimented with performing the intervention at different layers and reported the best performance in each case. We present the implementation details as follows.

TaskVec (Hendel et al., 2023): We first collect the task vectors using a few-shot setup where the few-shot examples contain the conflict evidence E_C along with the memorised answer M. We then follow the procedure illustrated in the original paper and extract the Task Vectors as the hidden representation of the last token (: in our case) at all layers. More specifically, we use the hidden representation of samples that generate answers following the context to create $\overline{\mathbf{h}_C}$, while we use those that follow the parameteric knowledge and ignore the context to create $\overline{\mathbf{h}_M}$. At inference time, we replace the residual stream at layer L with $\overline{\mathbf{h}_C}$ or $\overline{\mathbf{h}_M}$ to steer the model to follow the context or the parameteric knowledge, respectively.

ActAdd (Turner et al., 2023a): We collect the activations for ActAdd following the same procedure. For the inference-time intervention, we edit the residual stream by adding $\overline{\mathbf{h}_C}$ and subtracting $\overline{\mathbf{h}_M}$ to steer the model towards context knowledge, while we perform the opposite to steer the model towards parameteric knowledge.

SEA (Qiu et al., 2024): SEA adopts a different strategy and uses positive, negative and neutral model generations to compute steering vectors. We consider as neutral the generations that result from a few-shot setup where the demonstrations contain the memorised evidence E_M and memorised answer M, irrespective of the generated output. We then collect activations $\overline{\mathbf{h}_M}$ and $\overline{\mathbf{h}_C}$ following the same procedure illustrated above and use the method described in Qiu et al. (2024) to compute the steering vectors, assuming $\overline{\mathbf{h}_M}$ and $\overline{\mathbf{h}_C}$ encode the positive (follow parametric knowledge) and negative (follow context knowledge) behaviours respectively.

C.6 Searching Hyperparameters

We search hyperparameters of all methods using the same set of development sets. We choose hyperparameters based on the EM$_{C\rightarrow M}$ and EM$_{M\rightarrow C}$, which measure the capability of changing the behaviours of LLMs.

DoLa (Chuang et al., 2024): We search the best coefficient α that is used to compare premature and mature logits, evaluating with the "higher-layer" and "lower-layer" settings:

$$\log P(y) = \log P_{\text{mature}}(y) - \alpha \log P_{\text{premature}}(y).$$

We test the α ranging from -10 to 10 with an interval of 0.5. Finally, we set $\alpha = 6.0$ and $\alpha = -8.0$ to steer the model to use contextual and parametric knowledge for Llama2-7B and Llama3-8B; set $\alpha = 1.0$ and $\alpha = -1.0$ for Gemma2-9B. However, based on our experiments, though DoLa has a certain ability to change the behaviours of Gemma2-9B, we do not find a suitable α on both "high-layer"

and "low-layer" choices for improving Gemma2-9B on the overall performance measured by EM_M and EM_C.

CAD (Shi et al., 2024): We search the best coefficient α that is used to combine the logits with context and the logits without the context:

$$\log P(y) = (1+\alpha) \log P(y \mid c, x) - \alpha \log P(y \mid x).$$

We test the α ranging from -1.5 to 1.5 with an interval of 0.1. Finally, we set $\alpha = 0.6$ and $\alpha = -0.8$ to steer the model to use contextual and parametric knowledge for Llama2-7B and Llama3-8B; set $\alpha = 0.3$ and $\alpha = -0.3$ for Gemma-2-9B.

SPARE (Ours): Since no previous work used SAEs to control the knowledge selection behaviour of LLMs, we need first to identify suitable magnitudes of the hyperparameters and then search them in a smaller range in the development set. We fix the $\alpha = 1$ in Eq. (4) and test the proportion K to select the activations ranging from 0.1 to 0.01 with an interval of 0.01. Then, we choose $K = 0.07$ and test α ranging from 1.0 to 3.0 with an interval of 0.2. Finally, we select $K = 0.07$ and $\alpha = 2$ for Llama3-8B, $K = 0.06$ and $\alpha = 2.2$ for Llama2-7B. In our main experiments, SPARE edits 13, 14, 15, and 16 layers of Llama2-7B and Llama3-8B.

Due to the Gemma2-7B's SAEs (Lieberum et al., 2024; Rajamanoharan et al., 2024) using different training strategies and activation functions, we find they show a much more sparse pattern and have different suitable hyperparameters. For Gemma2-7B, we select and $K = 0.01$ and $\alpha = 3$ to steer to use contextual knowledge, and $K = 0.01$ and $\alpha = 1.8$ to steer to use parametric knowledge. We select the layers that have the highest knowledge detection accuracy to modify, which matches the practical requirements of inference time controlling. In our main experiments, SPARE edits 23, 24, 25, 29, 30 and 31 layers of Gemma2-9B.

We also present the selected top k activations in Appendix D for further analysis.

D Selected SAEs Activations of Gemma2-9B

We present the selected SAE activations used by SPARE for steering the knowledge selection behaviour. We can further interpret them in GemmaScope[5] (Lieberum et al., 2024).

[5] https://www.neuronpedia.org/gemma-scope

E Distribution of Mutual Information

In Appendix C.2, we mentioned that we use the proportion mutual information (K) to determine how many activations of the SAE (k) to select. Figs. 10 to 12 shows the layer-wise accumulated mutual information (x-axis) for the number of selected activations (y-axis) across different models (Gemma2-9B, Llama2-7B, Llama3-8B). In all three models, we observed that the graph is skewed when k (y-axis) is small, indicating that some SAE activations have relatively high mutual information. While there were some differences in this tendency between models (particularly pronounced in Gemma2-9B), we found only a little variation across selected layers within the same model. This analysis corresponds to the K values (from 0.01 for Gemma2-9B to 0.07 for Llama3-8B) that we identified through hyperparameter search in Appendix C.6.

F Distribution Patterns of the Residual Stream Under Knowledge Conflict

In this section, we provide further analysis of the representation patterns when knowledge conflicts in addition to Fig. 6b. Here, we focus on analysing the distribution patterns of different knowledge selection behaviours. More specifically, we compare the representation difference between the activation from $\mathcal{D}_C$ and $\mathcal{D}_M$, which are both under knowledge conflict but select contextual and parametric knowledge to generate the answer C and M. In Appendix F.1, we analyse the skewness patterns; in Appendix F.2, we analyse the L1 norm and L2 norm patterns.

F.1 Skewness of Residual Stream

In addition to Kurtosis, we used in Fig. 6b, we also measure the skewness by Hoyer and Gini index. We present the skewness patterns of hidden states in Fig. 13 and Fig. 14. We find the residual stream exhibits a significant skewed pattern when selecting the contextual knowledge to generate the answer. This observation supports the effectiveness of SPARE, where the residual steam becomes skewed when SPARE steers the model to generate contextual knowledge as shown in Fig. 6b.

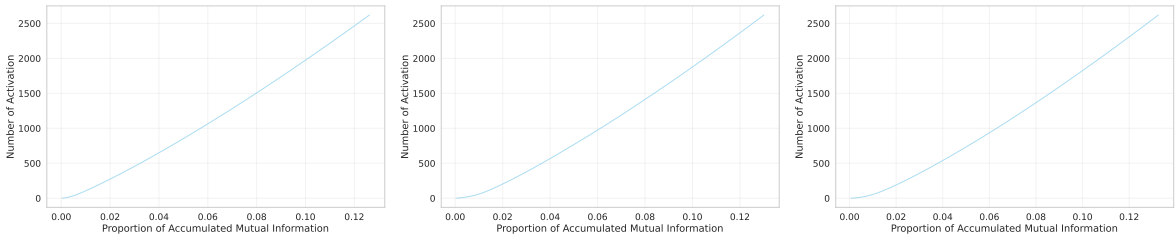
We also analyse the skewness pattern of MLP activations and Self-Attention activations in Fig. 15 and Fig. 16. However, we do not observe a distinct distribution difference like hidden states.

Layer	Use Parametric Knowledge	Use Contextual Knowledge
23	116391, 36331, 85142, 2795, 99547, 63615, 25635, 123378, 105328, 24132, 113025, 83008, 37706, 60782, 36046, 110864, 101469, 29902, 129485, 112858, 104185, 17911, 6673, 72533, 108414, 32967, 19761, 118260, 109917, 55083, 41965, 91874, 74605, 19726, 115338, 80100, 3042, 48088, 61830, 895, 49288, 120379, 105552, 84782, 14129	59646, 66244, 130943, 100165, 103568, 82090, 116937, 108558, 78302, 100628, 53091, 90600, 124049, 63656, 118525, 119623, 34458, 119574, 38170, 66293, 14026, 28797, 125520, 76467, 29583, 89951, 32901, 52256, 130987, 36816, 59062, 58505, 123631, 60183, 11432, 86969, 11755, 71200, 53746, 33, 57883, 67097, 108617, 112319, 1380, 47638, 42621, 16859, 130470, 6475, 112033, 101316, 40945, 82574, 58929, 79660, 81043, 18549, 4537, 130935, 127945, 78809
24	10649, 68997, 80242, 38885, 33450, 29004, 34725, 55203, 41474, 90933, 118013, 76436, 2795, 53138, 41501, 65408, 116855, 12056	76071, 55422, 82954, 40832, 68001, 88619, 120959, 92931, 38262, 83042, 42129, 21413, 74005, 73350, 57270, 6859, 83385, 9263, 8609, 22968, 8307, 99263, 2415, 59807, 87788, 92845, 88733, 124321, 25758, 111976, 84892, 104309, 61391, 60162, 128726, 28753, 62671, 80398, 40150, 28432, 81514, 9463
25	117145, 66103, 55992, 1609, 101788, 28707, 64494, 63602, 81174, 73438, 16428, 2054, 44642, 12418, 105769, 37692, 33693, 22786	77008, 39999, 65977, 3002, 82187, 113845, 35985, 16341, 121937, 13762, 9468, 70433, 42102, 85578, 3118, 99639, 41828, 58588, 103815, 70243, 67915, 125985, 113290, 127536, 84912, 2473, 46174, 100026, 37216, 27820, 81800, 13540, 125213, 79326, 55733, 32460, 46612
29	70665, 84563, 63717, 45653, 122282, 5001, 67756, 52905, 118450, 84589, 16721, 119640, 47070, 15218, 117432, 110719, 98957, 11667, 20824, 31422, 119807, 22664, 81261, 116958	65636, 113411, 88779, 19501, 46209, 8584, 71156, 79159, 94888, 144, 60280, 413, 103986, 74324, 52419, 70057, 30294, 13647, 37430, 71657, 118541, 12744, 74953, 115544, 19086, 102886, 49216, 95333, 26177, 89774, 71927, 70989, 23760
30	116964, 47548, 20615, 48375, 128786, 1308, 40865, 22211, 15816, 107813, 50419, 113319, 97588, 30688, 110627, 56882, 117785, 63602, 39609, 52155, 99243, 36852, 121514, 73310, 850, 96578	84358, 115174, 11363, 28696, 110664, 2831, 24365, 128820, 35092, 92968, 78722, 22739, 128047, 127030, 77294, 76467, 74131, 56766, 94697, 58000, 32812, 46910, 82749, 106077, 59596, 103936, 4505, 129363, 126847, 42463, 120310
31	61476, 5054, 1364, 18335, 63832, 88313, 35780, 130003, 25371, 125651, 11685, 24947, 2260, 70799, 92415, 47791, 99787, 88517, 85499, 75095, 114075, 125055, 109519, 116785, 100449, 37567, 88965, 59674, 14203, 125588, 70706, 18151	121514, 35148, 15479, 65369, 18623, 98225, 52746, 45804, 107893, 10202, 69463, 83810, 12131, 111417, 115174, 107085, 26328, 75203, 37430, 127639, 18114, 80704, 68360, 33142, 51607, 96802, 24949, 97568, 82042, 50826, 110615, 110929, 97833

Table 2: Selected Gemma2-9B SAEs activations with $K = 0.01$ (Eq. (5)). "Use Parametric Knowledge" means these activations are positively correlated with the behaviour of selecting parametric knowledge, determined according to our method described in Section 4.2.

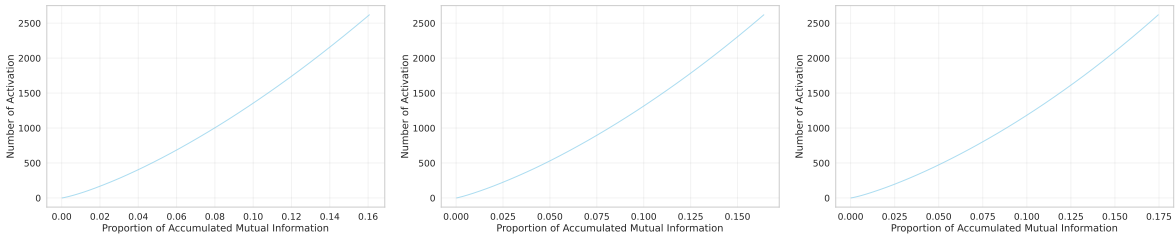
F.2 L1 Norm and L2 Norm Pattern

As we observe the distinct skewness pattern in hidden states, we further analyse their L1 norm and L2 norm patterns in Fig. 17 and Fig. 18 However, we do not observe the distinct norm differences between $\mathcal{D}_C$ and $\mathcal{D}_M$, though they have a significantly different skewness pattern.



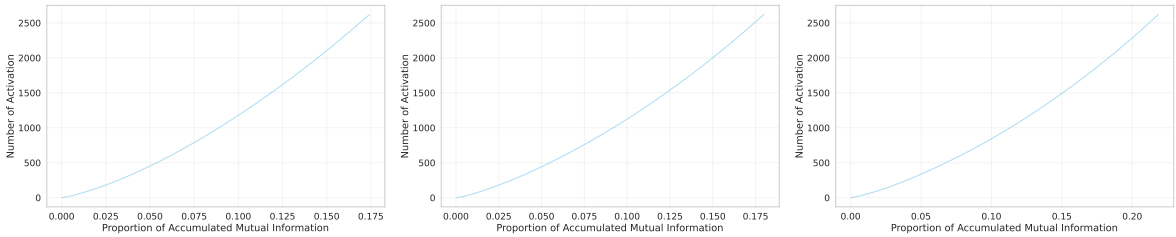
(a) Layer 23 (b) Layer 24 (c) Layer 25

Figure 10: Proportion of accumulated mutual Information (K) on Gemma2-9B



(a) Layer 13 (b) Layer 14 (c) Layer 15

Figure 11: Proportion of accumulated mutual Information (K) on Llama2-7B



(a) Layer 13 (b) Layer 14 (c) Layer 15

Figure 12: Proportion of accumulated mutual Information (K) on Llama3-8B

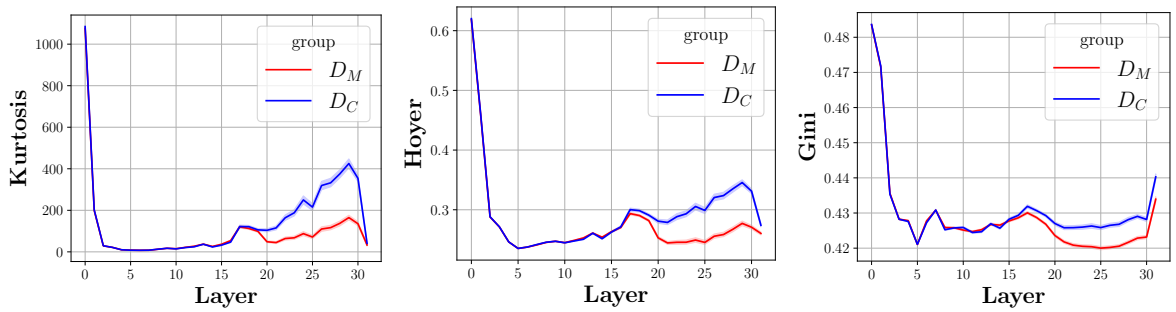


Figure 13: Skewness of the hidden states of Llama2-7B on NQSwap.

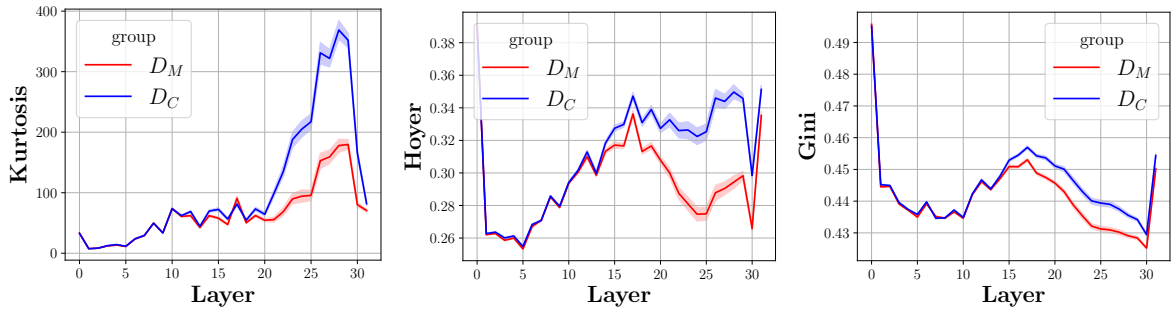


Figure 14: Skewness of the hidden states of Llama3-8B on NQSwap.

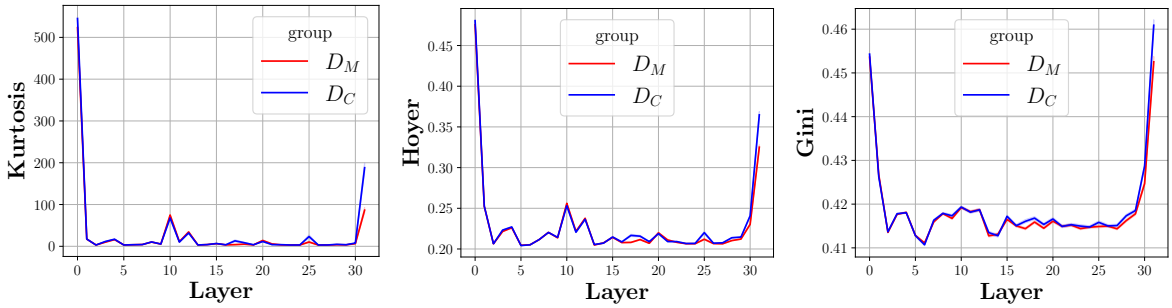


Figure 15: Skewness of the MLP activation of Llama2-7B on NQSwap.

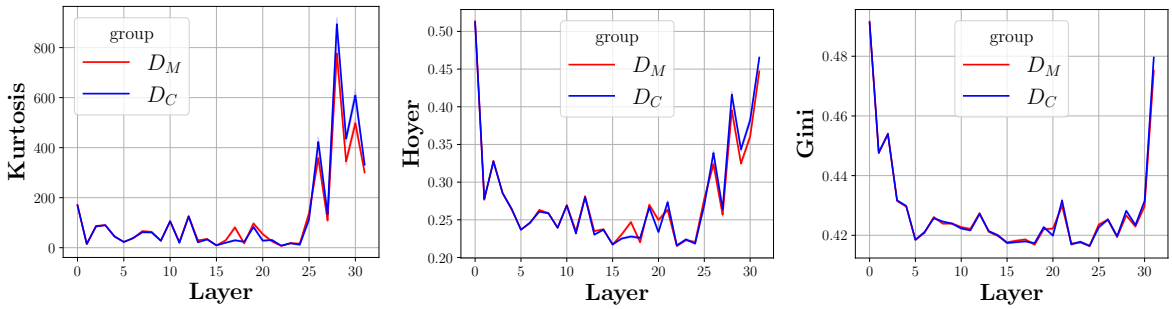


Figure 16: Skewness of the Self-Attention activation of Llama3-8B on NQSwap.

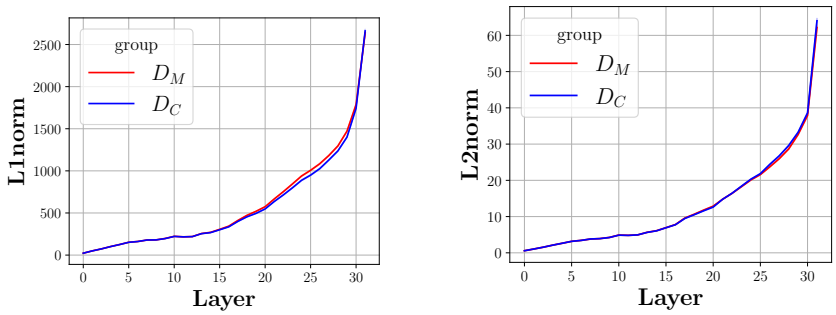


Figure 17: L1 norm and L2 norm of the hidden states of Llama3-8B on NQSwap.

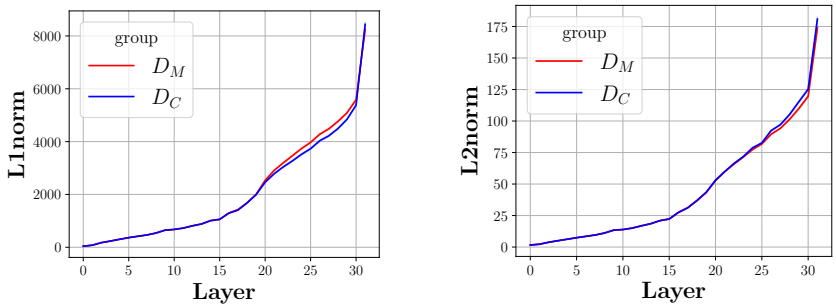


Figure 18: L1 norm and L2 norm of the hidden states of Llama2-7B on NQSwap.