Knowledge Circuit in Transformers

Anonymous ACL submission

Abstract

The remarkable capabilities of modern large language models are rooted in their vast repositories of knowledge encoded within their parameters, enabling them to perceive the world and engage in reasoning. The inner workings of how these models store knowledge have long been a subject of intense interest and investigation among researchers. To date, most studies have concentrated on isolated components within these models, such as the Multilayer Perceptrons and attention head. In this paper, we delve into the computation graph of the language model to uncover the knowledge circuits that are instrumental in articulating specific knowledge. The experiments, conducted with GPT2 and TinyLLAMA, has allowed us to observe how certain information heads, relation heads, and Multilayer Perceptrons collaboratively encode knowledge within the model. Moreover, we evaluate the impact of current knowledge editing techniques on these knowledge circuits, providing deeper insights into the functioning and constraints of these editing methodologies. Finally, we utilize knowledge circuits to analyze and interpret language model behaviors such as hallucinations and in-context learning. We believe the knowledge circuit holds potential for advancing our understanding of Transformers and guiding the improved design of knowledge editing.

1 Introduction

"Knowledge is power, and when embodied in the form of new technical inventions and mechanical discoveries it is the force that drives history." (bac; Bacon, 1949), Bacon's words are vividly re-enacted in the era of Large Language Models (LLMs) (OpenAI and the Co-authors, 2024; Zhao et al., 2023), as we witness their immense power in reshaping human society and redefining our understanding of machine intelligence. One thing that cannot be denied is that knowledge encapsulated within these models empowers their capabilities in reasoning, perceiving the world, and engaging in human-like communication. Nevertheless, these powerful models are not without their flaws. They still struggle with issues such as hallucinations (Huang et al., 2023; Wang et al., 2023a; Chen et al., 2024b), unsafe norms (Bonaldi et al., 2024; Sun et al., 2024), and offensive behaviors (Zhang et al., 2023; Jiang and Zubiaga, 2024) and these problems are exacerbated by the enigmatic internal mechanisms of knowledge storage within language models.

Recently, the research community has devoted significant efforts to unraveling the knowledge storage mechanisms of these models. Various studies (Dai et al., 2022a; Geva et al., 2023, 2022a, 2021a; Yu and Ananiadou, 2024; Chughtai et al., 2024; Meng et al., 2022; Merullo et al., 2024b) have been conducted to shed light on this intricate process, aiming to enhance our understanding and improve the safety and reliability of language models. The main finding in previous work is that knowledge may primarily stored in the Multilayer Perceptrons (MLPs) of Transformer-based language models. These MLPs function as a key-value neural memory, with knowledge being stored in what are termed "knowledge neurons" (KN). Based on these findings, researchers conduct Knowledge Editing (Meng et al., 2022; Yao et al., 2023) to update the language models' inaccurate facts, bias and unsafe content in their parametric space. Despite the initial success of these methods, there are still limitations, such as poor generalization, severe side effect, and failure to effectively utilize edited knowledge (Yao et al., 2023; Cohen et al., 2024), which motivate us to re-think previous approaches for interpreting knowledge storage in language models. Note that previous works treat the knowledge blocks as isolated components following the Restorative Theory (Hopkins, 2015), often focusing on identifying the specific blocks that stores particular knowledge. Several works (Niu et al., 2024; Zhang et al., 2024a) have proposed that different types of knowledge are often located in the same areas, suggesting that the current KN thesis may be an oversimplification.

To this end, instead of solely pinpointing tiny regions where the knowledge expressed can be localized, we aim to explore the cooperation between different components in Transformers like attention heads, MLPs, and embeddings, to understand how the language model stores and expresses the knowledge. Here, we introduce a new perspective: **Knowledge Circuit**, a critical subgraph in the language model to view the knowledge mechanism of Transformers. Note that Circuit, as a subgraph in the computation graph, has gained ever-growing attention in the mechanistic interpretability field (Elhage et al., 2021). Previous work (Wang

et al., 2023b; Merullo et al., 2024a) has found several important circuits for specific tasks like Indirect Object Identification and Color Object. These tasks necessitate the model to search the preceding context for a matching token and copy it into the next token prediction. However, in this work, we endeavor to construct a knowledge circuit for the knowledge that necessitates the model to utilize stored knowledge to make predictions, aiming to better unveil implicit neural knowledge representations, elucidate internal mechanisms for knowledge editing, and interpret more complex language model behaviors. Specifically, we leverage factual recall tasks and conduct experiments across various domains, including factual, bias, linguistic, and commonsense knowledge. We utilize GPT-2 (Radford et al., 2019) and TinyLLAMA (Zhang et al., 2024b) to explore the potential knowledge representations and utilization mechanism in these models. As shown in Figure 1 (a), we construct knowledge circuit associated with various expressions of knowledge using the existing knowledge stored in the language model. Through those discovered knowledge circuits, we find many interesting phenomena and conclusions as follows:

Knowledge circuit unveils implicit neural knowledge representations. We find that even when the knowledge circuit is used independently, the language model can recall related knowledge with a significant portion of its overall performance, demonstrating the effectiveness of those discovered knowledge representations (circuits). We also delve into specific pieces of knowledge and analyze the information flow within their respective circuits, indicating that language model tends to aggregate knowledge in the earlier to middle layers and further enhances this information in the later layers. We further uncover several special components (e.g., attention heads) in transferring information to the final token position and capturing relational information from the context (Figure 1 (b)).

Knowledge circuit elucidates internal mechanisms for knowledge editing. We conduct experiments to evaluate the impact of current knowledge editing methods on the language models' original knowledge circuits. Empirically, we observe that ROME (Meng et al., 2022) tends to incorporate edited information primarily at the edited layer. Subsequent mover heads (Appendix B.2) then transport this information to the residual stream of the last token. Conversely, during fine-tuning, the edited token is directly integrated into the language model, exerting a dominant influence on subsequent predictions.

Knowledge circuit facilitates interpreting language model behaviors. We further utilize the knowledge circuit to interpret language model behaviors, such as **hallucination** and **in-context learning**. We observe that when hallucination occurs, the language model fails to correctly transfer knowledge to the final token in the earlier layers. This is evident as the knowledge circuit lacks an effective "mover" head, or the mover head selects incorrect information. Additionally, we notice that several new attention heads emerge in the knowledge circuit during in-context learning.

2 Background: Circuit Theory

2.1 Preliminaries

In the context of neural network interpretability, a circuit can be conceptualized as a human-interpretable subgraph that is dedicated to executing specific tasks within a neural network model (Olah et al., 2020; Wang et al., 2023b; Lv et al., 2024; Conmy et al., 2023). When we visualize a neural network model as a connected directed acyclic graph (DAG), denoted as $\mathcal{G}$, the individual nodes represent the various components involved in the forward pass, such as neurons, attention heads, and embeddings. The edges symbolize the interactions between these components, including residual connections, attention mechanisms, and projections. A circuit, represented as $\mathcal{C} \subseteq \mathcal{G}$, emerges as a significant subgraph of $\mathcal{G}$ that is responsible for particular behaviors or functionalities. In this paper, we focus on the Transformer decoder architecture to conduct our experiments. The residual stream of Transformers has been demonstrated to be a valuable tool for mechanistic interpretability in recent works (Elhage et al., 2021; Yu and Ananiadou, 2024). The Transformer architecture typically starts with token embeddings, followed by a sequence of "residual blocks" and concludes with a token unembedding. Each residual block comprises an attention layer and an MLP layer, both of which "read" their input from the residual stream (by performing a linear projection) and "write" their output back to the residual stream through an additive projection. We can consider an attention head $A_{l,j}$ (the jth attention head in layer l) as operating on the residual stream from the previous layer, R_{l-1}. Given that $R_0 = I$ (where I represents the input embeddings), we can reinterpret attention head $A_{l,j}$ as processing the cumulative output of all previous attention heads and MLPs and input embedding, treating each node in the previous layers as separate input arguments. Similarly, an MLP node M_l can be seen as operating on the cumulative output of all previous attention heads and MLPs and input embedding, and the output node O operates on the sum of the input embeddings and the outputs of all attention heads and MLPs. The following equations represent the residual connections in the Transformer model, where R_l is the residual stream at layer l, and Input_l^A and Input_l^M are the inputs to the attention and MLP layers, respectively:

$$R_l = R_{l-1} + \sum_j A_{l,j} + M_l, R_0 = I$$

$$\text{Input}_l^A = I + \sum_{l'<l} \left(M_{l'} + \sum_{j'} A_{l',j'} \right)$$

$$\text{Input}_l^M = I + \sum_{l'<l} M_{i'} + \sum_{l'\leq i} \sum_{j'} A_{l',j'}$$

The computational graph $\mathcal{G}$ of the Transformer represents the interactions between attention heads and MLPs.

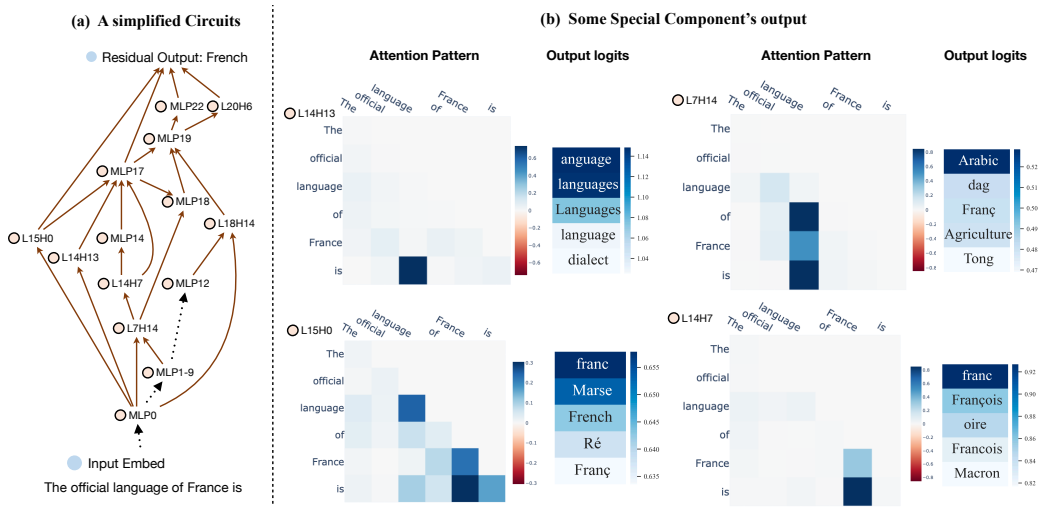


Figure 1: Knowledge circuit obtained from *"The official language of France is French"* in GPT2-Medium. Left: a simplified circuit and the whole circuit is in Figure 9 in Appendix. We use ⇢ to skip some complex connections between nodes. Right: the behavior of several special heads.

The nodes in $\mathcal{G}$ encompass the input embedding I, attention heads $A_{l,j}$, MLPs M_l, and the output node O, denoted as $N = \{I, A_{l,j}, M_l, O\}$. The edges in the model represent the connections between these nodes, $E = \{(n_x, n_y), n_x, n_y \in N\}$. A circuit $\mathcal{C}$ is meticulously constructed to govern specific behaviors within the model, comprising a selection of nodes $N_\mathcal{C}$ and edges $E_\mathcal{C}$ that are critical to the successful execution of the tasks at hand, expressed as $\mathcal{C} =< N_\mathcal{C}, E_\mathcal{C} >$.

2.2 Circuit Discovery

To identify circuits within a language model, a key approach is to examine the model's casual mediation by systematically altering the model's edges and nodes to observe the effects on performance (Conmy et al., 2023; Pearl, 2022; Vig et al., 2020). The underlying principle is that critical edges or nodes are those whose removal results in a notable decline in the model's predictive capabilities. Since the edges in the model's computational graph represent the dependencies between nodes, we can simulate the absence of a particular node-to-node dependency by ablating an edge in the graph. For example, ablating an edge from $A_{i',j'}$ to $A_{i,j}$ involves replacing the contribution of $A_{i',j'}$ in the input to attention head $A_{i,j}$ with zero (in the case of zero ablation) or with the mean value of head $A_{i',j'}$ (in the case of mean ablation). The process of identifying critical edges or nodes through ablation can be broken down into the following steps: i) Overwrite the value of the edge (n_x, n_y) with a corrupted value (either zero or mean ablation), ii) Perform a forward pass through the model with the altered graph, iii) Compare the output values of the modified model with those of the original model using a chosen metric S (detailed in Eq.1). If the performance change is below a predefined threshold τ, we can consider the edge non-critical and remove it to obtain a new subgraph $\mathcal{G}/(n_x, n_y)$. In addition to ablation-based methods, recent works have also explored the use of sparse auto-encoders (He et al., 2024; Cunningham et al., 2023) to identify circuits within language models. This approach involves training an auto-encoder to learn a sparse representation of the model's internal structure, which can help reveal the underlying circuitry responsible for specific behaviors or functionalities.

3 Knowledge Circuit Discovery in Transformers

3.1 Knowledge Circuit Construction

Unlike previous work (Dai et al., 2022a; Meng et al., 2022), which managed to find out the specific areas that store knowledge, we pay extra heed to the information flow that activates subsequent knowledge for answering questions. Similar to (Goldowsky-Dill et al., 2023; Wang et al., 2023b), we write language model as a graph consisting of the input, the output, attention heads, and MLPs by considering a "residual rewrite" of the model's computational structure. For example, this residual rewrite gives us a nearly-dense graph in GPT2-medium: one between every pair of (attention head, MLP, input, and output) nodes, except for attention heads in the same layer, which do not communicate with each other. In our paper, we concentrate on the task of answering factual open-domain questions, where the goal is to predict a target entity o given a subject-relation pair (s, r). A knowledge triplet $k = (s, r, o)$ is often presented to the model in the form of a natural language prompt for next token prediction (e.g., *"The official language of France is* ____*"*). The model $\mathcal{G}$ is expected to generate the target entity, which is consistent with the language model's pretraining format. To identify the circuit that are critical for predicting the target entity o for a given subject-relation pair (s, r), we ablate each special edge $e_i = (n_x, n_y)$ in the computation graph $\mathcal{G}$. We then measure the impact of ablating the edge

(**zero ablation** in our implementation) on the model's performance using the MatchNLL loss (Conmy et al., 2023) for the target o:

$$S(e_i) = -\log(\mathcal{G}/e_i(o|(s,r))) - \log(\mathcal{G}(o|(s,r))) \quad (1)$$

If the score $S(e_i)$ is less than the predefined threshold τ, we consider the edge to be non-critical and remove it from the computation graph, updating the temporary circuit $\mathcal{C}_{temp} \leftarrow \mathcal{G}/e_i$. We first sort the graph by topological rank following Conmy et al. (2023) and traverse all edges in this manner, We derive a circuit $\mathcal{C}_k$ that contributes to representing the knowledge necessary to answer the factual question:

$$\mathcal{C}_k =< N_k, E_k > \quad (2)$$

Here, $\mathcal{C}_k$ is the circuit for the knowledge triplet k, consisting of the nodes N_k and edges E_k that are essential for predicting the target entity o given the subject-relation pair (s, r).

3.2 Knowledge Circuit Information Analysis

Once we have identified the knowledge circuit, we delve deeper into the specific roles and behaviors of each node and edge within the computation graph. Our goal is to comprehend the processing and contribution of each node n_i to the functionality of the circuit. Drawing on the methodologies of previous studies (Yu and Ananiadou, 2024; Katz and Belinkov, 2023; nostalgebraist, 2020a), we begin by applying layer normalization to the output of each node n_i and then map it into the embedding space. This is achieved by multiplying the layer-normalized output by the unembedding matrix ($\mathbf{W}_U$) of the language model: $\mathbf{W}_U \,\mathrm{LN}(n_i)$. This transformation allows us to inspect how each component writes information to the circuit and how it influences subsequent computational steps. By understanding the nodes' behavior in the circuit, we can better comprehend the circuit's structure and the key points where information is aggregated and disseminated.

3.3 Knowledge Circuit Experimental Settings

Implementations. We conduct experiments on GPT-style models, including GPT-2 medium and large. We also conduct primary experiments on TinyLLaMA (Zhang et al., 2024b) to validate the effectiveness of different architectures. We utilize the Automated Circuit Discovery (Conmy et al., 2023) toolkit to build a circuit as an initiative of our analysis and leverage transformer lens (Nanda and Bloom, 2022) to further analyze the results. Specially, we simply employ the MatchNLL (Conmy et al., 2023) as the metric to detect the effect of the given node and edge and use **zero ablation** to knock out the specific computation node in the model's computation graph.

Metrics. A discovered knowledge circuit is deemed an accurate representation of a specific area within the transformer's knowledge storage, thus, it should be capable of representing the knowledge independently. Following (Conmy et al., 2023), we leverage the completeness of a circuit which refers to its ability to independently reproduce the behavior or predictions of the full model for the relevant tasks. This property is assessed by examining whether the identified subgraph corresponds to the underlying algorithm implemented by the neural network. To evaluate completeness, we first construct the circuit using the validation data D_{val} for a specific knowledge type and then test its performance on the test split D_{test} in isolation. By doing so, we can observe any changes in performance compared to the original model. We use the **`Hit@10`** metric to measure the rank of the target entity o among the top 10 predicted tokens:

$$\textbf{Hit@10} = \frac{1}{|V|} \sum_{i=1}^{|V|} \mathrm{I}\left(\mathrm{rank}_o \leq 10\right) \quad (3)$$

Here, $|V|$ represents vocabulary size, and rank_o is the rank of the target entity o in predictions.

Dataset. In this work, we focus on the **knowledge that already stored in the language model**. We utilize the dataset provided by LRE (Hernandez et al., 2024) and considered different kinds of knowledge, including linguistic, commonsense, fact, and bias. We evaluate whether the knowledge is present in the language model's parameters under zero-shot settings using the **`Hit@10`** metric to sample knowledge from the validation set, which is used to construct the knowledge circuit. The data statistics are in Appendix A.

4 Knowledge Circuit Unveils Implicit Neural Knowledge Representations

Knowledge Circuit Evaluation. We report the results of GPT2-Medium in Table 1, wich indicates that with only less than 10% of the original knowledge circuit's subgraph, the model can maintain over 70% of its original performance. One of the most fascinating observations is the **performance improvement seen on several test datasets**. For instance, the *Landmark-country* relation metric increases from 0.16 to 0.36. This suggests that the discovered knowledge circuits may encapsulate the relevant knowledge, and the model's performance on these tasks could have been hindered by noise from other components. We proceed to analyze the layer distribution of the original model $\mathcal{G}$ to understand the average percentage of nodes that are activated within the circuit for different knowledge domains. From Figure 2, we observe that attention and MLPs are more active in the lower layers of the network, where the model processes the input and extracts general information. To gain a more comprehensive view of the information processing, we compute the rank_o change of the target token across the layers and we show the results in Figure 6. This analysis reveals the phenomenon of early decoding (nostalgebraist, 2020a), suggesting that by the middle to the latest layers, the target entity is already present in the residual stream, and the subsequent layers in the Transformer are designed to increase the proba-

Table 1: **Hit@10** of the Original and Circuit Standalone performance of knowledge circuit in GPT2-Medium. **The result for D_{val} being 1.0 indicates that we select the knowledge for which the model provides the correct answer to build the circuit.**

Type	Knowledge	#Edge	D_{val}		D_{test}	
			Original($\mathcal{G}$)	Circuit($\mathcal{C}$)	Original($\mathcal{G}$)	Circuit($\mathcal{C}$)
Linguistic	Adj Antonym	573	0.80	1.00 ↑	0.00	0.40 ↑
	world first letter	432	1.00	0.88	0.36	0.16
	world last letter	230	1.00	0.72	0.76	0.76
Commonsense	object superclass	102	1.00	0.68	0.64	0.52
	fruit inside color	433	1.00	0.20	0.93	0.13
	work location	422	1.00	0.70	0.10	0.10
Factual	Capital City	451	1.00	1.00	0.00	0.00
	Landmark country	278	1.00	0.60	0.16	0.36 ↑
	Country Language	329	1.00	1.00	0.16	0.75 ↑
	Person Native Language	92	1.00	0.76	0.50	0.76 ↑
Bias	name religion	423	1.00	0.50	0.42	0.42
	occupation age	413	1.00	1.00	1.00	1.00
	occupation gender	226	1.00	0.66	1.00	0.66
	name birthplace	276	1.00	0.57	0.07	0.57 ↑
Avg			0.98	0.73	0.44	0.47 ↑

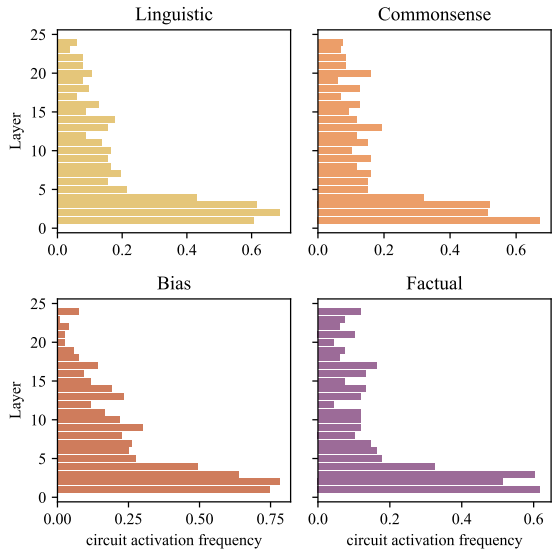


Figure 2: The activated circuit component distributions in Layers in GPT2-Medium.

bility of the current token (See discussion in the running example).

Special Components in Knowledge Circuit. From the discovered knowledge circuit, we can find several important attention heads that demonstrate specific behavior, including the **mover head** (Lv et al., 2024), **relation head** (Chughtai et al., 2024; Ferrando et al., 2024) and **mixture head** (Chughtai et al., 2024; Ferrando et al., 2024). These details would be accumulated by the MLP in the model and we would discuss the behavior of these special heads in the running example part. We list some of these special components in Table 3 in Appendix. The different attention heads are responsible for expressing specific types of knowledge and may be activated by different facts. In our experiments with GPT-2 Medium and GPT-2 Large, we find that knowledge is distributed across several layers' attention heads and MLP matrices, suggesting that the target knowledge appears to have been accumulated throughout the GPT-2 model. Conversely, in TinyLLAMA, the critical components are more concentrated. As depicted in Figure 6, the rank of the target entity in TinyLLAMA experiences a sharp decline around several layers, whereas in the GPT2 model, the decline is more gradual. We hypothesize that this discrepancy may be attributed to the model's knowledge capacity (Allen-Zhu and Li, 2024) and warrants further investigation.

A Running Example of Knowledge Circuit. We present a case and analyze the specific behavior of components within the identified knowledge circuit. Taking the factual knowledge *"The official language of France is French"* as an example, we visualize the knowledge circuit in Figure 1. To visualize the information flow within the model more effectively, we have plotted the rank and probability of the target entity o at each layer when it is mapped into the embedding space, in Figure 3. After MLP 17, the target knowledge emerges as the top token in the residual stream and after that layer, it undergoes an increased probability. The edges connected to MLP17 are (L14H13 → MLP17), (L14H7 → MLP17), and (L15H0 → MLP17) . Here, the L14H13 is a relation head that focuses on the relation token in the context. The output of this head is relation-related tokens such as *"language"* and *"Language"*. The attention head L14H7 is a mover head that moves the

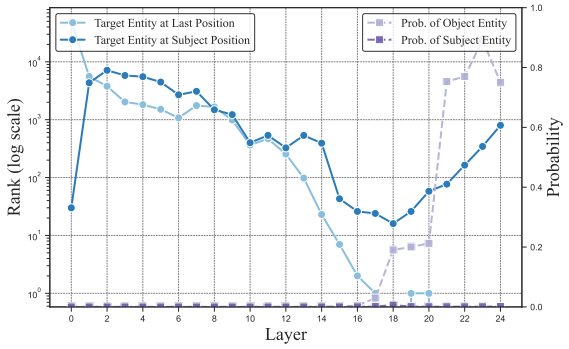


Figure 3: The rank and probability of the target entity o when unembedding the intermediate layer's output for the fact *"The official language of France is French"*. For a more comprehensive view, we additionally plot the probability of the subject entity at the last token position and the target entity's rank at the subject position.

information from the subject position *"France"* to the last token. Previous work (Lv et al., 2024; Merullo et al., 2024b) have introduced this mover head as an *argument parser*, which moves *"France"* to the last token, and the MLP conducts a function application to map *"France"* to *"French"*. An intriguing observation is that we can find the output of this head already contains the target entity, which significantly contributes to the final output (L14H7 $\rightarrow$ Output). Also, we see the probability of the subject token at the last token is nearly zero across these layers. Hence, instead of the *argument parser*, we consider this mover head as an *extract head* proposed by Geva et al. (2023), which aims to extract the related-information from the subject token's position. In the subsequent knowledge editing experiments, we can observe changes in the behavior of these types of heads. Instead of extraction in the later layers proposed by Geva et al. (2023), we notice a gradual decrease in rank across all early-to-middle layers. The MLP17 combines information from previous tokens and integrates this information to prioritize the target token at the top rank. Interestingly, upon tracing the information flow to L14H7, we discovered that it is predominantly activated by L7H14, a relation head, and its output features several language tokens, such as *"Arabic"*. We hypothesize that L7H14 may function as a signaling mechanism to activate the associated mover head, but this hypothesis necessitates further investigation to be confirmed. After MLP17, several attention heads, such as L18H14 (a relation head) and L20H6 (a mover head), collaborated to further enhance the final prediction of the target entity.

5 Knowledge Circuit Elucidates Internal Mechanisms for Knowledge Editing

In this section, our objective is to evaluate the impact of previous knowledge editing methods (why fail in certain cases and settings) and validate the effectiveness of knowledge circuits.

Single Factual Knowledge Editing. Here, we adopt the ROME method (Meng et al., 2022) and FT-M (Zhang et al., 2024a), which aim to edit the MLP layers in the language model. The most important hyperparameter in knowledge editing is the layer, as the same method's performance varies significantly via the layers. Here, we evaluate the performance of different editing layers and their effectiveness. We compare the knowledge circuit computed by the edited model with the original one, and we present results in Figure 4. For details about the edited knowledge circuit, please refer to Appendix D. As discussed in the previous part, the early-to-middle layers are the main part of aggregating the target entity o to the top rank. In the original model, the probability of the target entity *"Intel"* is nearly zero, and the model fails to elevate it to the top rank in the vocabulary. Edit layer 0 with ROME and FT-M both give us the correct answer but we can view a different scenario for their knowledge circuits. For **ROME**, as the correct information is added to the subject position, we can recognize a **behaviour change of the Mover Head changes from copying to extracting the edited information from the subject position**. This information aggregates gradually through the following layers and at layer 15, the *"Intel"* emerges as the top rank and undergoes increased probability. Specially, before editing, the mover head L15H3 attends to the *"controller"* token and returns *"controller"* as the output, while in the edited model, the attention head's output moves to the *"Intel"*, which means the model gains the information at the subject space. While in the FT-M edit, the edited model tends to directly write the knowledge into the specific component, which would greatly dominate the following component in the model. As shown in Figure 4, the output logits in MLP-0 for *"Intel"* are more than 10, and it emerges as the top rank in the residual stream directly. This phenomenon can be found in different knowledge types and layers and we report results in Appendix D.2. However, the added knowledge may have the risk to influence unrelated knowledge. When we test another fact *"Windows server"*, the model still tends to give us the *"Intel"* answer, demonstrating the overfitting problem. This finding supports previous analysis regarding the correlation between localization and editing (Hase et al., 2023), suggesting that edits may not alter the storage but merely add signal into the knowledge circuit.

Multi-hop Factual Knowledge Editing. Multi-hop knowledge editing poses a challenging scenario (Yao et al., 2023; Cohen et al., 2024; Zhong et al., 2023), wherein we edit the model with new knowledge, yet the model struggles to perform reasoning using the edited information. We analyze multi-hop questions in language models (Yang et al., 2024; Ju et al., 2024) to understand why current editing methods fail in these scenarios. For instance, given the fact (Thierry Mugle, *"home country"*, France), we edit the fact to another country, such as (Thierry Mugle, *"home country"*, France

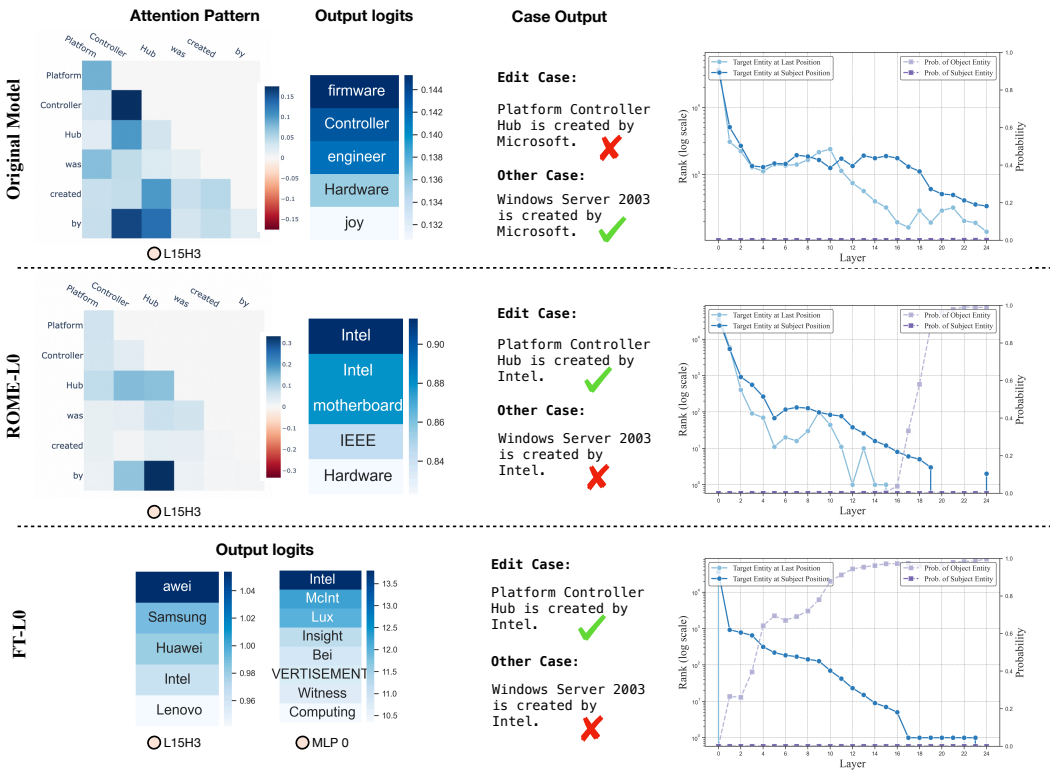


Figure 4: Different behaviors when we edit the language model. In the original model, we can see the mover head L15H3 actually move the original token *"Controller"* and other information, while for ROME, we observe the mover head select the correct information *"Intel"*, which means ROME **successfully added the *"Intel"* to model**. For the FT layer-0 editing, we can find this method **directly write the edited knowledge into edited component**. However, we find these two editing methods would affect other unrelated input *"Windows server is created by?"*

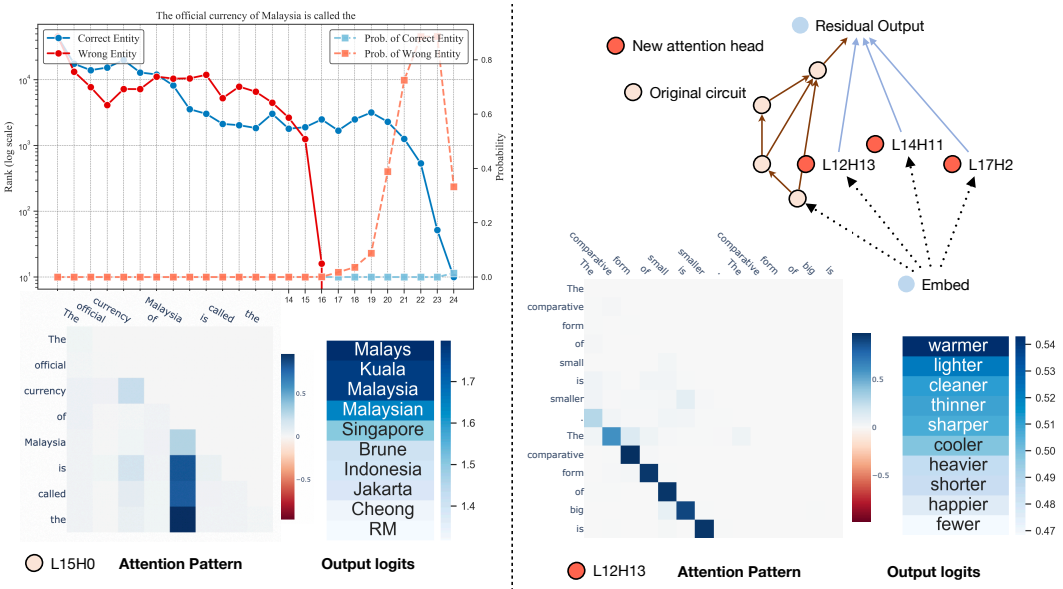
→ China). We then assess the model's performance on questions based on the edited knowledge, including *"The official currency of the home country of Thierry Mugle is"* and *"The capital city of the home country of Thierry Mugle is"*. While the unedited model could correctly answer these questions, we observe that the edited model would provide the answer *"China"* for subsequent hop reasoning. We find that the mover head in the original multi-hop reasoning circuit initially extracts the second-hop answer but, after editing, extracts *"China"*, demonstrating that the edited information dominantly saturates and influences the circuit. Furthermore, we observe an intriguing phenomenon: **even in the original model's multi-hop reasoning settings, it would directly provide the answer if we removed the context of the first-hop texts** (Details in Appendix C.1). This further confirms the findings that the model relies on relational and subject-related information, regardless of grammatical adherence.

6 Knowledge Circuit Facilitates Interpreting Language Model Behaviors

In this Section, our aim is to validate whether the identified knowledge circuit is actually utilized by the model when it employs knowledge. To address this, as shown in Figure 5, we investigate three phenomena: hallucination, in-context learning, and reverse relations (Details in Appendix C.3).

Factual Hallucination. We focus on factual hallucinations, which occur when the model provides an incorrect target entity for a given subject s and relation r. In our experiments (Figure 5 and Appendix C.2), we observe that the model fails to move the correct knowledge to the final token in the earlier layers. This failure is evident as the circuit lacked an effective mover head or the mover head selected incorrect information. For instance, in the prompt *"The official currency of Malaysia is called"*, both the correct answer *"Ringgit"* and the incorrect one *"Malaysian"* were accumulated before layer 15. However, at layer 16, the mover head L15H10 extracted the erroneous information. Despite a rank drop of the true one in layers 20–22, this is insufficient to correct the previous mistake.

In-Context Learning. Despite storing vast amounts of knowledge, a language model may still provide incorrect answers. However, with demonstrations or examples (based on RAG (Gao et al., 2023)), it can quickly generate correct responses. In this section, we focus on the scenario where the model initially provides an incorrect answer but can then produce the correct response upon receiving the appropriate demonstration. We con-



(a) A hallucination Case

(b) An in-context learning Case

Figure 5: Left: fact hallucination case *"The official currency of Malaysia is called"*, we can find **at the layer 15 the mover head select the wrong information**. Right: In-context learning case, we can see **some new heads that focus on the demonstration appear** in the zero-shot knowledge circuit.

sider the original knowledge circuit and introduce a new knowledge circuit based on the demonstration. Our analysis reveals that, compared to the zero-shot knowledge circuit, several new attention heads appear in the computation graph when the demonstration is incorporated as shown in Figure 5. These heads mainly focus on the demonstration's context: *"The comparative form of small is smaller"*. Concretely, Todd et al. (2024) have identified a concept known as the Function Vector, which represents the average of some key attention heads. In our experiments, we found that these attention heads primarily focus on the demonstration, indicating their role in leveraging the demonstration to reactivate and correct the model's response.

7 Related Work

Knowledge Mechanism of Transformers. How the language model store and utilize knowledge is an ongoing research topic. Previous works find that the MLP in Transformers works as a key-value memory and stores enormous knowledge (Dai et al., 2022a; Geva et al., 2021a, 2022a; Meng et al., 2022). As to the relation between entities, Hernandez et al. (2024) observe that facts can be linearly decoded from the enriched subject residual stream by mapping the subject entity to the object entity. Instead of viewing the knowledge storage in isolation, Geva et al. (2023); Lv et al. (2024); Yu and Ananiadou (2024) find the knowledge is accumulated during the layers. Regarding knowledge analysis, Bayazit et al. (2023) also attempt to discover critical knowledge in language models. However, they only consider several layers in the model and use the pruning method, which may overlook the connections between components. More related works can be found in Appendix E.1.

Manipulate Language Models. Recently, many works have aimed to manipulate the language models to make the model aligned with world knowledge or social value norms, such as knowledge editing (Yao et al., 2023; Zhang et al., 2024a), machine unlearning (Si et al., 2023; Chen and Yang, 2023) and detoxification (Wang et al., 2024). Most of these works are elicited by previous knowledge mechanism findings such as knowledge neuron (Dai et al., 2022b). They modify the MLP in the LLM (Meng et al., 2022; Dai et al., 2022a) to change the model's behavior based on specific factual knowledge. However, recent works (Li et al., 2023a; Sakarvadia et al., 2023) demonstrate the pivotal role of the attention part in knowledge representation. Hase et al. (2023) also observe that the performance of editing in a layer may not be a reliable way to localize the fact. In this paper, we manipulate specific knowledge of language model via knowledge circuit, including both MLP and attention components across different layers.

8 Conclusion

In this paper, we offer a new perspective on viewing knowledge storage based on circuit theory and conduct a primary analysis to demonstrate its effectiveness. We hope these findings can advance our understanding of the knowledge mechanisms of language models and provide insights for better designing and editing language models, enhancing knowledge, and improving reasoning to enhance factuality and alleviate hallucinations.

Limitations and Broader Impacts

Current circuit discovery-based patching method is time-consuming, there's some concurrent work that proposes more efficient way (Ferrando and Voita, 2024) to build the model's information flow. Also, there are some other methods to discover circuits, like acdcpp (Syed et al., 2023) and Sparse Auto-Encoders (Bricken et al., 2023; He et al., 2024). We believe that knowledge circuit discovery has a huge room for improvement. Additionally, by focusing on linguistic, factual, commonsense, and bias-related knowledge, we believe our approach can be applied to ensure safety and privacy information to promote trustworthy AI.

References

Francis bacon. https://iep.utm.edu/francis-bacon/.

Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebron, and Sumit Sanghai. 2023. GQA: Training generalized multi-query transformer models from multi-head checkpoints. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 4895–4901, Singapore. Association for Computational Linguistics.

Zeyuan Allen-Zhu and Yuanzhi Li. 2024. Physics of language models: Part 3.3, knowledge capacity scaling laws.

Francis Bacon. 1949. The advancement of learning [1605]. In *Primer of intellectual freedom*, pages 172–192. Harvard University Press.

Deniz Bayazit, Negar Foroutan, Zeming Chen, Gail Weiss, and Antoine Bosselut. 2023. Discovering knowledge-critical subnetworks in pretrained language models. *Preprint*, arXiv:2310.03084.

Beren and Sid Black. 2022. The singular value decompositions of transformer weight matrices are highly interpretable. https://www.lesswrong.com/posts/mkbGjzxD8d8XqKHzA/the-singular-value-decompositions-of-transformer-weights.

Lukas Berglund, Meg Tong, Maximilian Kaufmann, Mikita Balesni, Asa Cooper Stickland, Tomasz Korbak, and Owain Evans. 2024. The reversal curse: LLMs trained on "a is b" fail to learn "b is a". In *The Twelfth International Conference on Learning Representations*.

Helena Bonaldi, Yi-Ling Chung, Gavin Abercrombie, and Marco Guerini. 2024. Nlp for counterspeech against hate: A survey and how-to guide. *Preprint*, arXiv:2403.20103.

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

Jiaao Chen and Diyi Yang. 2023. Unlearn what you want to forget: Efficient unlearning for LLMs. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12041–12052, Singapore. Association for Computational Linguistics.

Shiqi Chen, Miao Xiong, Junteng Liu, Zhengxuan Wu, Teng Xiao, Siyang Gao, and Junxian He. 2024a. In-context sharpness as alerts: An inner representation perspective for hallucination mitigation. *Preprint*, arXiv:2403.01548.

Xiang Chen, Chenxi Wang, Yida Xue, Ningyu Zhang, Xiaoyan Yang, Qiang Li, Yue Shen, Lei Liang, Jinjie Gu, and Huajun Chen. 2024b. Unified hallucination detection for multimodal large language models. *CoRR*, abs/2402.03190.

Yuheng Chen, Pengfei Cao, Yubo Chen, Kang Liu, and Jun Zhao. 2024c. Journey to the center of the knowledge neurons: Discoveries of language-independent knowledge neurons and degenerate knowledge neurons. In *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*, pages 17817–17825. AAAI Press.

Yuheng Chen, Pengfei Cao, Yubo Chen, Yining Wang, Shengping Liu, Kang Liu, and Jun Zhao. 2024d. The da vinci code of large pre-trained language models: Deciphering degenerate knowledge neurons. *CoRR*, abs/2402.13731.

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*.

Bilal Chughtai, Alan Cooney, and Neel Nanda. 2024. Summing up the facts: Additive mechanisms behind factual recall in llms. *Preprint*, arXiv:2402.07321.

Roi Cohen, Eden Biran, Ori Yoran, Amir Globerson, and Mor Geva. 2024. Evaluating the Ripple Effects of Knowledge Editing in Language Models. *Transactions of the Association for Computational Linguistics*, 12:283–298.

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

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

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. 2022a. Knowledge neurons in pretrained transformers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8493–8502, Dublin, Ireland. Association for Computational Linguistics.

Damai Dai, Li Dong, Yaru Hao, Zhifang Sui, Baobao Chang, and Furu Wei. 2022b. Knowledge neurons in pretrained transformers. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 8493–8502. Association for Computational Linguistics.

Fahim Dalvi, Hassan Sajjad, and Nadir Durrani. 2023. Neurox library for neuron analysis of deep NLP models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics: System Demonstrations, ACL 2023, Toronto, Canada, July 10-12, 2023*, pages 226–234. Association for Computational Linguistics.

Subhabrata Dutta, Joykirat Singh, Soumen Chakrabarti, and Tanmoy Chakraborty. 2024. How to think step-by-step: A mechanistic understanding of chain-of-thought reasoning. *CoRR*, abs/2402.18312.

Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg, and Christopher Olah. 2022. Toy models of superposition. *CoRR*, abs/2209.10652.

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. 2021. A mathematical framework for transformer circuits. *Transformer Circuits Thread*. Https://transformer-circuits.pub/2021/framework/index.html.

Mostafa Elhoushi, Akshat Shrivastava, Diana Liskovich, Basil Hosmer, Bram Wasti, Liangzhen Lai, Anas Mahmoud, Bilge Acun, Saurabh Agarwal, Ahmed Roman, et al. 2024. Layer skip: Enabling early exit inference and self-speculative decoding. *arXiv preprint arXiv:2404.16710*.

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

Javier Ferrando and Elena Voita. 2024. Information flow routes: Automatically interpreting language models at scale. *arXiv preprint arXiv:2403.00824*.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Qianyu Guo, Meng Wang, and Haofen Wang. 2023. Retrieval-augmented generation for large language models: A survey. *CoRR*, abs/2312.10997.

Mor Geva, Jasmijn Bastings, Katja Filippova, and Amir Globerson. 2023. Dissecting recall of factual associations in auto-regressive language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12216–12235, Singapore. Association for Computational Linguistics.

Mor Geva, Avi Caciularu, Kevin Wang, and Yoav Goldberg. 2022a. Transformer feed-forward layers build predictions by promoting concepts in the vocabulary space. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 30–45, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Mor Geva, Avi Caciularu, Kevin Ro Wang, and Yoav Goldberg. 2022b. Transformer feed-forward layers build predictions by promoting concepts in the vocabulary space. 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 30–45. Association for Computational Linguistics.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021a. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021b. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 5484–5495. Association for Computational Linguistics.

Asma Ghandeharioun, Avi Caciularu, Adam Pearce, Lucas Dixon, and Mor Geva. 2024. Patchscopes: A unifying framework for inspecting hidden representations of language models. *Preprint*, arXiv:2401.06102.

Nicholas Goldowsky-Dill, Chris MacLeod, Lucas Sato, and Aryaman Arora. 2023. Localizing model behavior with path patching. *Preprint*, arXiv:2304.05969.

Peter Hase, Mohit Bansal, Been Kim, and Asma Ghandeharioun. 2023. Does localization inform editing? surprising differences in causality-based localization vs. knowledge editing in language models. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Zhengfu He, Xuyang Ge, Qiong Tang, Tianxiang Sun, Qinyuan Cheng, and Xipeng Qiu. 2024. Dictionary learning improves patch-free circuit discovery in mechanistic interpretability: A case study on othello-gpt. *Preprint*, arXiv:2402.12201.

Evan Hernandez, Arnab Sen Sharma, Tal Haklay, Kevin Meng, Martin Wattenberg, Jacob Andreas, Yonatan Belinkov, and David Bau. 2024. Linearity of relation decoding in transformer language models. In *Proceedings of the 2024 International Conference on Learning Representations*.

Belinda Hopkins. 2015. *Restorative theory in practice: Insights into what works and why*. Jessica Kingsley Publishers.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. 2023. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *Preprint*, arXiv:2311.05232.

Aiqi Jiang and Arkaitz Zubiaga. 2024. Cross-lingual offensive language detection: A systematic review of datasets, transfer approaches and challenges. *Preprint*, arXiv:2401.09244.

Zhuoran Jin, Pengfei Cao, Hongbang Yuan, Yubo Chen, Jiexin Xu, Huaijun Li, Xiaojian Jiang, Kang Liu, and Jun Zhao. 2024. Cutting off the head ends the conflict: A mechanism for interpreting and mitigating knowledge conflicts in language models. *CoRR*, abs/2402.18154.

Tianjie Ju, Yijin Chen, Xinwei Yuan, Zhuosheng Zhang, Wei Du, Yubin Zheng, and Gongshen Liu. 2024. Investigating multi-hop factual shortcuts in knowledge editing of large language models. *Preprint*, arXiv:2402.11900.

Shahar Katz and Yonatan Belinkov. 2023. VISIT: Visualizing and interpreting the semantic information flow of transformers. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 14094–14113, Singapore. Association for Computational Linguistics.

Kenneth Li, Oam Patel, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. 2023a. Inference-time intervention: Eliciting truthful answers from a language model. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Maximilian Li, Xander Davies, and Max Nadeau. 2023b. Circuit breaking: Removing model behaviors with targeted ablation. *Workshop on Challenges in Deployable Generative AI at International Conference on Machine Learning*.

Ang Lv, Kaiyi Zhang, Yuhan Chen, Yulong Wang, Lifeng Liu, Ji-Rong Wen, Jian Xie, and Rui Yan. 2024. Interpreting key mechanisms of factual recall in transformer-based language models. *Preprint*, arXiv:2403.19521.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in GPT. *Advances in Neural Information Processing Systems*, 36.

Kevin Meng, Arnab Sen Sharma, Alex J. Andonian, Yonatan Belinkov, and David Bau. 2023. Mass-editing memory in a transformer. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Jack Merullo, Carsten Eickhoff, and Ellie Pavlick. 2024a. Circuit component reuse across tasks in transformer language models. In *The Twelfth International Conference on Learning Representations*.

Jack Merullo, Carsten Eickhoff, and Ellie Pavlick. 2024b. Language models implement simple word2vec-style vector arithmetic. *Preprint*, arXiv:2305.16130.

Dan Mossing, Steven Bills, Henk Tillman, Tom Dupré la Tour, Nick Cammarata, Leo Gao, Joshua Achiam, Catherine Yeh, Jan Leike, Jeff Wu, and William Saunders. 2024. Transformer debugger. `https://github.com/openai/transformer-debugger`.

Neel Nanda and Joseph Bloom. 2022. Transformerlens. `https://github.com/neelnanda-io/TransformerLens`.

Jingcheng Niu, Andrew Liu, Zining Zhu, and Gerald Penn. 2024. What does the knowledge neuron thesis have to do with knowledge? In *The Twelfth International Conference on Learning Representations*.

nostalgebraist. 2020a. interpreting GPT: the logit lens.

nostalgebraist. 2020b. interpreting gpt: the logit lens. `https://www.lesswrong.com/posts/AcKRB8wDpdaN6v6ru/interpreting-gpt-the-logit-lens`.

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

OpenAI and the Co-authors. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

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

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Mansi Sakarvadia, Aswathy Ajith, Arham Khan, Daniel Grzenda, Nathaniel Hudson, André Bauer, Kyle Chard, and Ian Foster. 2023. Memory injections: Correcting multi-hop reasoning failures during inference in transformer-based language models. In *Proceedings of the 6th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages

342–356, Singapore. Association for Computational Linguistics.

Nianwen Si, Hao Zhang, Heyu Chang, Wenlin Zhang, Dan Qu, and Weiqiang Zhang. 2023. Knowledge unlearning for llms: Tasks, methods, and challenges. *Preprint*, arXiv:2311.15766.

Lichao Sun, Yue Huang, Haoran Wang, Siyuan Wu, Qihui Zhang, Chujie Gao, Yixin Huang, Wenhan Lyu, Yixuan Zhang, Xiner Li, Zhengliang Liu, Yixin Liu, Yijue Wang, Zhikun Zhang, Bhavya Kailkhura, Caiming Xiong, Chaowei Xiao, Chunyuan Li, Eric P. Xing, Furong Huang, Hao Liu, Heng Ji, Hongyi Wang, Huan Zhang, Huaxiu Yao, Manolis Kellis, Marinka Zitnik, Meng Jiang, Mohit Bansal, James Zou, Jian Pei, Jian Liu, Jianfeng Gao, Jiawei Han, Jieyu Zhao, Jiliang Tang, Jindong Wang, John Mitchell, Kai Shu, Kaidi Xu, Kai-Wei Chang, Lifang He, Lifu Huang, Michael Backes, Neil Zhenqiang Gong, Philip S. Yu, Pin-Yu Chen, Quanquan Gu, Ran Xu, Rex Ying, Shuiwang Ji, Suman Jana, Tianlong Chen, Tianming Liu, Tianyi Zhou, William Wang, Xiang Li, Xiangliang Zhang, Xiao Wang, Xing Xie, Xun Chen, Xuyu Wang, Yan Liu, Yanfang Ye, Yinzhi Cao, and Yue Zhao. 2024. Trustllm: Trustworthiness in large language models. *CoRR*, abs/2401.05561.

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

Eric Todd, Millicent Li, Arnab Sen Sharma, Aaron Mueller, Byron C Wallace, and David Bau. 2024. LLMs represent contextual tasks as compact function vectors. In *The Twelfth International Conference on Learning Representations*.

Eric Todd, Millicent L. Li, Arnab Sen Sharma, Aaron Mueller, Byron C. Wallace, and David Bau. 2023. Function vectors in large language models. *CoRR*, abs/2310.15213.

Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart Shieber. 2020. Investigating gender bias in language models using causal mediation analysis. *Advances in neural information processing systems*, 33:12388–12401.

Cunxiang Wang, Xiaoze Liu, Yuanhao Yue, Xiangru Tang, Tianhang Zhang, Cheng Jiayang, Yunzhi Yao, Wenyang Gao, Xuming Hu, Zehan Qi, Yidong Wang, Linyi Yang, Jindong Wang, Xing Xie, Zheng Zhang, and Yue Zhang. 2023a. Survey on factuality in large language models: Knowledge, retrieval and domain-specificity. *Preprint*, arXiv:2310.07521.

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

Mengru Wang, Ningyu Zhang, Ziwen Xu, Zekun Xi, Shumin Deng, Yunzhi Yao, Qishen Zhang, Linyi Yang, Jindong Wang, and Huajun Chen. 2024. Detoxifying large language models via knowledge editing. *Preprint*, arXiv:2403.14472.

Peng Wang, Ningyu Zhang, Xin Xie, Yunzhi Yao, Bozhong Tian, Mengru Wang, Zekun Xi, Siyuan Cheng, Kangwei Liu, Guozhou Zheng, et al. 2023c. Easyedit: An easy-to-use knowledge editing framework for large language models. *arXiv preprint arXiv:2308.07269*.

Wenhao Wu, Yizhong Wang, Guangxuan Xiao, Hao Peng, and Yao Fu. 2024. Retrieval head mechanistically explains long-context factuality. *Preprint*, arXiv:2404.15574.

Sohee Yang, Elena Gribovskaya, Nora Kassner, Mor Geva, and Sebastian Riedel. 2024. Do large language models latently perform multi-hop reasoning? *Preprint*, arXiv:2402.16837.

Yunzhi Yao, Peng Wang, Bozhong Tian, Siyuan Cheng, Zhoubo Li, Shumin Deng, Huajun Chen, and Ningyu Zhang. 2023. Editing large language models: Problems, methods, and opportunities. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 10222–10240, Singapore. Association for Computational Linguistics.

Zeping Yu and Sophia Ananiadou. 2024. Locating factual knowledge in large language models: Exploring the residual stream and analyzing subvalues in vocabulary space. *Preprint*, arXiv:2312.12141.

Mert Yuksekgonul, Varun Chandrasekaran, Erik Jones, Suriya Gunasekar, Ranjita Naik, Hamid Palangi, Ece Kamar, and Besmira Nushi. 2024. Attention satisfies: A constraint-satisfaction lens on factual errors of language models. In *The Twelfth International Conference on Learning Representations*.

Junlin Zhang. 2023. Parametric reflection of the world: Why can gpt generate intelligence through next token prediction. https://zhuanlan.zhihu.com/p/632795115.

Ningyu Zhang, Yunzhi Yao, Bozhong Tian, Peng Wang, Shumin Deng, Mengru Wang, Zekun Xi, Shengyu Mao, Jintian Zhang, Yuansheng Ni, et al. 2024a. A comprehensive study of knowledge editing for large language models. *arXiv preprint arXiv:2401.01286*.

Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. 2024b. Tinyllama: An open-source small language model. *Preprint*, arXiv:2401.02385.

Zhexin Zhang, Leqi Lei, Lindong Wu, Rui Sun, Yongkang Huang, Chong Long, Xiao Liu, Xuanyu Lei, Jie Tang, and Minlie Huang. 2023. Safetybench: Evaluating the safety of large language models with multiple choice questions. *CoRR*, abs/2309.07045.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, Yifan Du, Chen Yang, Yushuo Chen, Zhipeng Chen, Jinhao

Category	# Rel.	# Examples	# GPT-2 Corr.
Factual	26	9,696	4,721
Commonsense	8	374	240
Linguistic	6	806	483
Bias	7	213	149

Table 2: Information about the dataset. Evaluation is always restricted to the subset of (s, r, o) triples for which the LM successfully decodes o when prompted with (s, r). Table borrowed from Hernandez et al. (2024)

Jiang, Ruiyang Ren, Yifan Li, Xinyu Tang, Zikang Liu, Peiyu Liu, Jian-Yun Nie, and Ji-Rong Wen. 2023. A survey of large language models. *CoRR*, abs/2303.18223.

Zexuan Zhong, Zhengxuan Wu, Christopher Manning, Christopher Potts, and Danqi Chen. 2023. MQuAKE: Assessing knowledge editing in language models via multi-hop questions. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 15686–15702, Singapore. Association for Computational Linguistics.

Appendix

A Implementation Details

Hyper-parameter. The primary hyperparameter for constructing a circuit is the threshold τ used to detect performance drops. Setting τ too high may result in an incomplete circuit, while setting it too low can introduce numerous unnecessary nodes. In our experiment, we tested τ values from the set {0.02, 0.01, 0.005} to determine the appropriate circuit size for different types of knowledge.

Implementation. We utilize ACDC[1] to construct circuits that encode specific knowledge representations. Additionally, we re-implement the code to compute the relevant dataset and impact metrics for the knowledge used. Since TinyLLAMA[2] incorporates the Grouped Query Attention Mechanism (Ainslie et al., 2023), we interleave and repeat the key and value pairs to analyze the specific behavior of each attention head. We use the NVIDIA-A800 (40GB) to conduct our experiments. It took about 1-2 days to compute the circuit for the knowledge type in GPT2-medium.

Dataset Details. All the data used in our paper is sourced from Hernandez et al. (2024), with detailed information provided in Table 5. In the original setting, they use the data for few-shot settings, but in our experiments, we considered zero-shot knowledge storage, so here we sample the data using the **`Hit@10`** to detect whether the model understands knowledge for the given prompt based on the s and o. We sample the test set in a 1:1 ratio with the validation set to ensure a balanced evaluation.

[1] https://github.com/ArthurConmy/Automatic-Circuit-Discovery

[2] checkpoint: https://huggingface.co/TinyLlama/TinyLlama-1.1B-intermediate-step-1431k-3T

B More Experiment Results

B.1 Rank Change Across Layers

To gain a clearer understanding of the knowledge aggregation phenomenon, we computed the rank of the target entity rank_o within the vocabulary space $|V|$ at the output of each layer. As depicted in Figure 6, we observe that the model initially elevates the target entity to the top ranks of the vocabulary. Once the entity reaches the top of the vocabulary, subsequent layers continue to enhance its probability mass. These findings corroborate previous work by (Chen et al., 2024a; Chuang et al., 2024; Elhoushi et al., 2024), who noted the substantial difference in logit entropy between layers, contributing to the model's improved prediction for the target entity. In our work, we aim to delve deeper into how the model, as well as specific components within it, give rise to these behaviors.

B.2 Special Components in Knowledge Circuit

When zooming into the discovered circuit, we can find several kinds of special attention heads, or MLPs, in the model that play a pivotal role in the final prediction, similar to what previous research has indicated (Lv et al., 2024). We follow previous work (Lv et al., 2024; Chughtai et al., 2024) and list the definition of these heads as follows:

- **Mover Head** (Lv et al., 2024; Merullo et al., 2024a): These heads focus on the last token of the context and attend to the subject token, functioning as a mover to transfer information.
- **Relation Head** (Chughtai et al., 2024): These heads would attend to the relation token in the context and produce some relation-related tokens that would guide the behaviour of the following components.
- **Mix Head** (Chughtai et al., 2024; Ferrando et al., 2024): These head would focus on both the relation token and the subject tokens. In our experiment, we found these heads usually work similar as the mover head.

Lv et al. (2024) posits that these mover heads are responsible for extracting the "argument" from the context and passing it for further processing, such as function application. Consequently, Lv et al. (2024); Merullo et al. (2024b) suggests that the MLP within the language model performs a function application that transforms the subject into an object, with the subject's probability ranking higher than the object's. However, these findings are limited to specific cases, as the studies only examined two related tasks, such as capital city identification or color objection. Our analysis suggests that these conclusions may not be universally applicable to all knowledge domains and require further investigation. When we examine the ranks of subject and object entities, we rarely observe an overwhelming superiority at the last token position of the subject token and usually,

the probability of the subject token at the last position is uniformly low. In our experiment, we view the model as a collaboration between different nodes in the identified circuit. They perform as different kind of attention heads, like *mover head* and *relation head*. We list some heads that are responsible for the storage of different kind of knowledge and relations in Table 3.

Component Reuse Phenomenon. Merullo et al. (2024a) identified shared circuits for the IOI task and the Colored Objects task. We also observe this phenomenon in the factual recall task. As depicted in Table 3, we can observe that for related relations such as "city_in_country", "name_birth_place", and "country_language", their circuits include both L21H12, which stores and maps country-related information. Additionally, we found that some *relation heads* are activated by different relations. For instance, in our experiments, the head 'L7H14' appears in the circuits of both "official_language" and "official_currency". We speculate that these reused heads, rather than task-specific heads, can be considered as topic heads, as proposed by (Beren and Black, 2022; Ferrando and Voita, 2024). We believe that further investigation into this distinction is warranted in future research.

Table 3: Critical component behaviour in circuits as task-specific head. Find more results in Appendix

Model	Type	Fact	Critical Component in Circuit
GPT2-Medium	Linguistic Factual Commonsense Bias	Antonym city country work location name country	L17H2, L18H1, L13H12, L13H8 L21H12, L16H2 L19H15, L14H4, L13H3 L16H6, L21H12
GPT2-Large	Linguistic Factual Commonsense Bias	Antonym company hq work location name country	L25H5, L24H16, L19H13, L18H8 L30H6, L25H13 L18H13, L28H18, L30H5 L21H19, L29H2
TinyLLAMA	Linguistic Factual Commonsense Bias	Verb past tense Landmark country Fruit Inside Color name country	L17H0, MLP20 L15H11, L17H19, MLP18 L18H25, MLP18 L15H11, MLP17

B.3 A Case on TinyLLAMA

In this part, we demonstrate one circuit we found in the TinyLAMA in Figure 7. Actually, in TinyLLAMA, the attention heads bearing specific behavors in the later layers is usually less than GPT2. We can also view some mover heads and relation heads in the circuit that would generate the target token as the output such as the L15H3 and L17H19.

C More Circuit Utilization Analysis

While previous work on knowledge storage suggests that knowledge may be localized in specific areas of the model, it is essential to ascertain whether the model actively employs this knowledge when encountering related contexts or if it relies on shortcuts.

C.1 Multi-hop Factual Knowledge Editing

We considered scenarios where the model makes a correct prediction for all the multi-hop questions and single-hop questions. We found a circuit reuse phenomenon in the one-hop and multi-hop knowledge circuits. Here, we first compute the proportion of the nodes N_{single} in the single-hop circuit C_{single} that appears in $N_{multiple}$ the set of nodes in the multi-hop circuit $C_{multiple}$.

Table 4: Hit for different hop

	First-hop	Second-hop	Integrate
node	83.33	70.27	71.42
edge	63.20	45.27	49.42

$$Hit_{node} = \frac{|N_{multiple}| \cap |N_{single}|}{|N_{single}|} \quad (4)$$

Actually, there are two ways for the model to conduct multi-hop reasoning. As shown in the figure, the model can also answer the question by combining the two hop relations together (in the given case, combine "hometown" and "language" as "mother tongue") If the model is capable of combining two-hop relations in a more integrated or semantic way, such as inferring that the "mother tongue" is the language of one's hometown, this suggests a more complex reasoning process that goes beyond the simple overlap of nodes and edges. To capture this kind of reasoning, we assesses the model's ability to integrate information from different hops. $R_{integrated}$ as the set of integrated relations (new paths created by combining information from different hops) We observed an overlap of these circuit nodes, indicating that the language model utilizes a large portion of the nodes in the original single hop's circuit, especially the mover head. From the overlap analysis, it seems the model utilizes single-hop information to conduct reasoning. However, we discover an intriguing phenomenon in GPT-2 and TinyLLAMA: the models can correctly answer first-hop knowledge and perform multi-hop reasoning based on it. However, interestingly, when we delete the first-hop knowledge while retaining only the second-hop relation and the first-hop subject, the models can still correctly answer the multi-hop question. Figure 8 illustrates a specific case in our findings. It further enhances our previous findings that the model actually conducts the factual recall with the relation head and the gathered information about the subject. **Moreover, we found this phenomena is hugely alleviated by the aligned model, which requires further investigation in the future.**

C.2 Hallucination

The results are presented in Figure 5. we observe an interesting phenomenon: the correct answer and the wrong answer are both accumulated at the previous layer, but at some specific layers, the wrong answer is selected as the answer to extract. We hypothesize that there may be a **circuit competition** here proposed by (Zhang, 2023) and we detect the behavior of the specific component between them.

C.3 Reverse Relation

Reverse Curse (Berglund et al., 2024) is an important issue in language models when models successfully give us the correct answer o for (s, r), but with a reverse

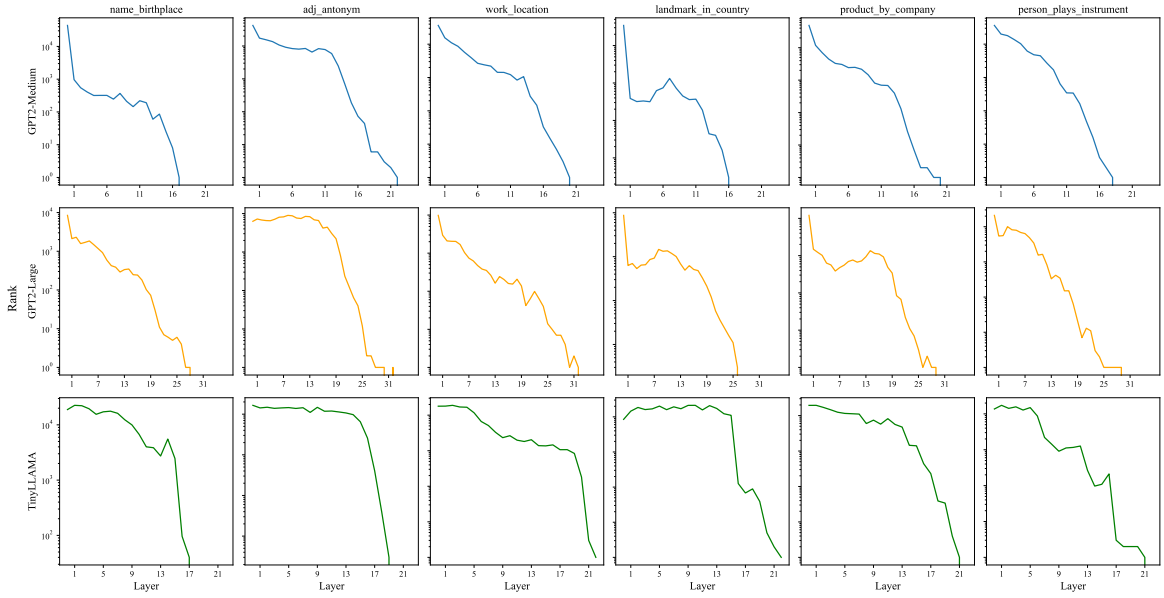


Figure 6: The average rank of the target entity o for the D_{val} in the vocabulary when mapping the output of each layer in the model to the embedding space. We can find that the in GPT2-Medium and GPT2-Large, the model would get the knowledge at middle-to-later layers. While for TinyLLAMA, the layer may be more later.

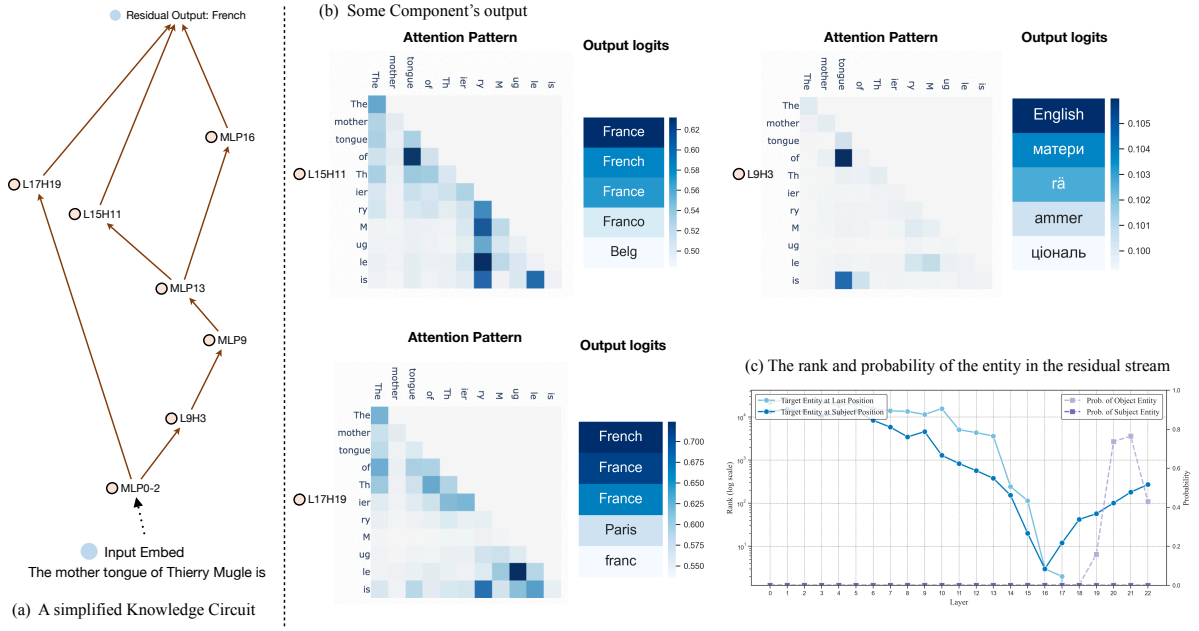


Figure 7: A simplified knowledge circuit found in TinyLLAMA for the knowledge *"The mother tongue of Thierry Mugle is French"*.

relation $\hat{r}$ and o, the models fail to give us the correct subject s. In this part, we endeavor to investigate how the LM manages the reverse relation when they successfully store the knowledge. We first sample facts where the LM successfully predicts the given knowledge and the reversed fact. Then, we compute the overlap between these two circuits, $\mathcal{C}_d$ and $\mathcal{C}_r$ under node levels based on equation 4. We select the *"superhero_person"* relation to see the difference of these two circuit and test the node overlap of the two circuit in the model. We notice that the overlap between the two circuits is about 70%, indicating the language model may store the related information in the same place. We also found the activated mover heads for the two relationships to be identical.

D Edit Experiments

D.1 Method Implementation

ROME As proposed by Meng et al. (2022), ROME view editing as a minimal optimization problem. ROME

Table 5: Number of examples per relation and the count of accurate predictions by different LMs. This table is borrowed from Hernandez et al. (2024) and here we sampled with different ways.

Category	Relation	#	# Correct in Hit@10		
			GPT2-Medium	GPT2-large	TinyLLaMA
factual	person mother	994	83	144	361
	person father	991	359	385	474
	person sport position	952	400	489	596
	landmark on continent	947	835	543	694
	person native language	919	310	220	260
	landmark in country	836	600	489	709
	person occupation	821	57	76	149
	company hq	674	308	312	470
	product by company	522	422	432	460
	person plays instrument	513	510	505	498
	star constellation name	362	266	148	297
	plays pro sport	318	317	316	315
	company CEO	298	20	52	125
	superhero person	100	28	35	50
	superhero archnemesis	96	6	6	23
	person university	91	14	37	35
	pokemon evolution	44	11	13	16
	country currency	30	25	25	30
	food from country	30	23	25	29
	city in country	27	20	23	27
	country capital city	24	24	24	24
	country language	24	24	24	24
	country largest city	24	24	24	24
	person lead singer of band	21	7	16	21
	president birth year	19	11	12	-
	president election year	19	17	18	-
commonsense	object superclass	76	62	64	72
	word sentiment	60	14	9	34
	task done by tool	52	44	45	45
	substance phase of matter	50	12	16	48
	work location	38	28	24	27
	fruit inside color	36	36	35	36
	task person type	32	28	27	26
	fruit outside color	30	16	20	21
linguistic	word first letter	241	236	235	241
	word last letter	241	135	73	114
	adjective antonym	100	80	81	84
	adjective superlative	80	24	19	63
	verb past tense	76	1	15	76
	adjective comparative	68	7	15	63
bias	occupation age	45	18	20	18
	univ degree gender	38	14	35	38
	name birthplace	31	29	30	31
	name religion	31	24	31	31
	characteristic gender	30	26	30	30
	name gender	19	19	19	19
	occupation gender	19	19	19	19

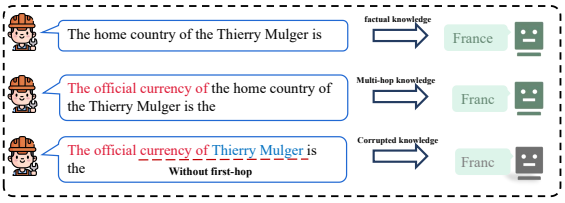


Figure 8: a specific case in Multi-hop reasoning. When we removed the context of the first hop question, we found the model also directly gave us the answer. The phenomenon appears in both GPT2 and TinyLLAMA.

conceptualizes the MLP module as a straightforward key-value store. Specifically, the key represents a subject and the value encapsulates knowledge about that subject, the MLP can reestablish the association by retrieving the corresponding value for the key. To add a new key-value pair, ROME applies a rank-one modification to the MLP's weights, effectively "writing in" the new information directly. This method enables more precise and direct modification of the model's knowledge. The ROME method for model editing was conducted based on the EasyEdit(Wang et al., 2023c) framework, utilizing the default parameters provided by EasyEdit. The experiments were performed on an A800 80G GPU,

with approximately 8GB of memory consumption.

FT-M For Fine-Tuning (FT-M), we follow Zhang et al. (2024a). It trains the MLP layer using the cross-entropy loss on the target answer while masking the original text. This approach aligns more closely with the traditional fine-tuning object. The FT method for model editing was similarly conducted using the EasyEdit[3] (Wang et al., 2023c) framework, with the default parameters provided by EasyEdit. The experiments were also performed on an A800 80G GPU, with memory consumption of approximately 10GB.

D.2 Edit Cases on FT-M and ROME

When a circuit is established for a particular piece of knowledge, we can manipulate the model's computation by targeting critical points within the circuit. Li et al. (2023b) ablates a small number of important causal pathways by masking the edges in the circuit and making the model less toxic and safer, which proves the effectiveness of the circuit. As illustrated in figure 11 and figure 12, we present the changes in the ranking of the predicted probabilities for the target new token when editing layer 6, 12, and 18 of the GPT-2 medium model using FT and ROME methods. When applying FT for model editing, it is evident that the rank of the target new token's probability sharply declines at the corresponding edited layer, resulting in a vertical line in the figure. This indicates that FT directly embeds the editing information into the model's information flow. Conversely, when using the ROME method for editing, this effect is mitigated. The predicted probability of target new token reaches its peak only a few layer after the edited layer. This observation is consistent with our previous analysis in section 5.

E More related Work and Discussion

E.1 More Related Work

Knowledge in MLP Geva et al. (2021b) claim that MLP serve as key-value memories for knowledge in LMs. Geva et al. (2022b) further propose the knowledge neuron theory, suggesting that the key and value vectors in MLPs encode factual knowledge. Based on the above findings, Chen et al. (2024c) discover that multiple distinct sets of KNs can store identical facts, Chen et al. (2024d) explore the structural and functional among neurons by neurological topology clustering method. Meng et al. (2022) and Meng et al. (2023) confirm that MLP modules do store factual knowledge and pioneering use knowledge editing methods to modify outdated knowledge stored in LM. Anthropic recently introduces scaling monosemanticity[4], which extracts highly abstract features that respond to and behaviorally cause abstract behaviors.

[3] https://github.com/zjunlp/EasyEdit

[4] https://transformer-circuits.pub/2024/scaling-monosemanticity/index.html

Knowledge in Attention Heads Li et al. (2023a) reveal that some attention heads are capable of truthful answer. Wu et al. (2024) investigate 4 model families, 6 model scales, and 3 types of finetuning, and find retrieval heads, which are responsible for retrieving relevant information from long context. Jin et al. (2024) suggest that memory heads can recall knowledge from internal memory and context heads can retrieve knowledge from external context. Todd et al. (2023) use causal mediation analysis on a diverse range of in-context-learning and find some attention heads, dubbed function vectors, which trigger the ability of in-context-learning.

Knowledge in Hybrid Components Recent works emphasize the importance of connections of components among LM for knowledge representation and utilization. Geva et al. (2023) describe factual recall by three steps: (1) subject enrichment in MLP sublayers, as in ROME (Meng et al., 2022), (2) relation propagation to the END token, and (3) selective attribute extraction by later layer attention heads. Lv et al. (2024) apply projection and intervention to explore mechanisms in factual recalls tasks, and conclude that task-specific attention head moves the topic entity to the final position of the residual stream, while MLP conducts relation function.

Circuit Circuit discovery also plays a significant role in analyzing the internal mechanisms of the entire model (Elhage et al., 2021). Specifically, a circuit, comprising components such as MLP and attention, is a subgraph of the computation graph. Conmy et al. (2023) design an automated circuit discovery approach that implements the specified behavior. Wang et al. (2023b) explain the circuits for Indirect Object Identification (IOI) task. They use causal interventions to discover circuits responsible for the flow of information. Instead the above task-specific circuit, Merullo et al. (2024a) present evidence a circuit is shared by similar tasks in IOI and Color Object (CO) tasks. Dutta et al. (2024) construct circuits using attention heads, and further observe that attention heads are pivotal to chain-of-thought reasoning, i.e., attention heads move information along ontological relationships appear exclusively in the initial half layers, write the answer token predominantly appear in the later half layers. However, the above-mentioned circuit studies either focus solely on a single component (MLP or attention) or only explore IOI and CO tasks. IOI and CO tasks necessitate the model to search the preceding context for a matching token and copy it into the next token prediction. Also, despite the success of previous circuit discovery, we can hardly make it into real usage. Hence, in this work, we attempt to analyze knowledge circuit consisting of both MLP and attention components, and investigate the current editing methods' effect on the circuit to shed light on the future.

Tools There are also many tools that are designed to analyze the LM's behavior, such as Logit Lens (nos-

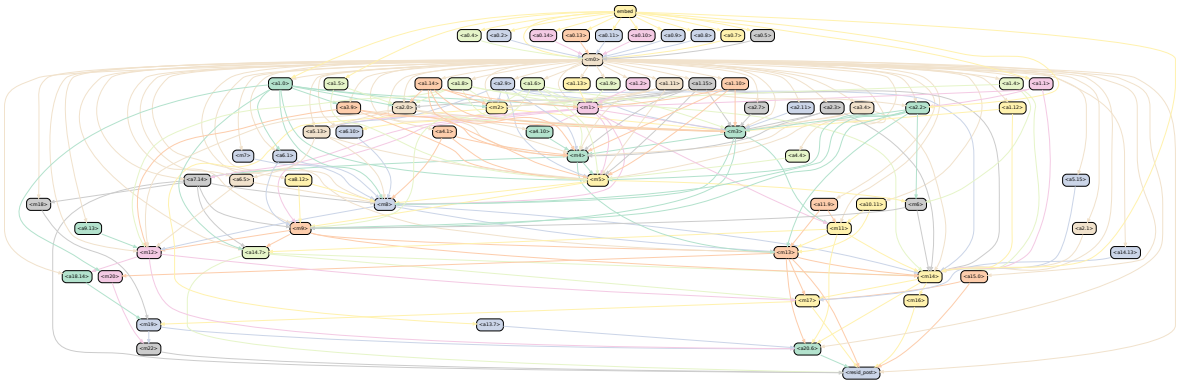
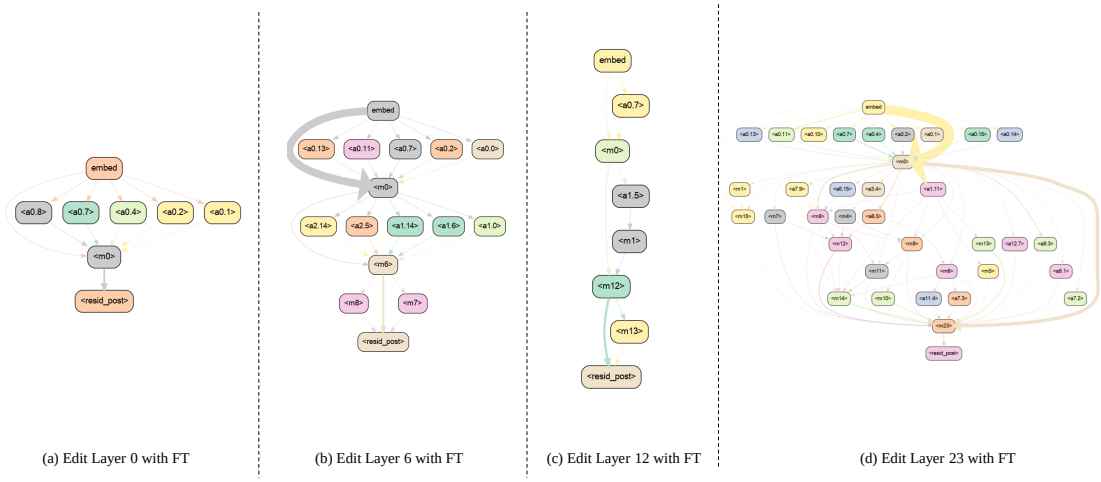


Figure 9: The knowledge circuit from the *"The official language of France is French"* in GPT2-Medium.



(a) Edit Layer 0 with FT (b) Edit Layer 6 with FT (c) Edit Layer 12 with FT (d) Edit Layer 23 with FT

Figure 10: The knowledge circuit obtained from the edited model for the case *"Platform Controller Hub was created by"* with the target entity "Intel" shows that when editing the model using different layers, the fine-tuned settings allow the edited MLP to directly provide the edited information.

talgebraist, 2020b), Attention len (Yuksekgonul et al., 2024), Attribution lens (Hernandez et al., 2024) and transformer-lens (Nanda and Bloom, 2022). NeuroX (Dalvi et al., 2023) implements various interpretation methods under a unified API and provides insights into how knowledge is structured in representations and discovers the role of neurons in LM. Transformer Debugger (Mossing et al., 2024) is an intepretability tool provided by OpenAI, which deploys the GPT-4 and sparse auto-encoder to explain the language neurons and attention head. PatchScope (Ghandeharioun et al., 2024) is a tool provided by Google which use a new model to explain the hidden states in the original model.

E.2 Limitation and Future Discussion

Despite of the attempt to combine the attention head and MLP to view the knowledge storage as a whole, this work operates with a relatively coarse granularity of circuits. For instance, the neurons within a MLP may necessitate a finer level of granularity to fully capture their behavior and contributions. Even we now know these component work together to express the knowledge, why they are activated are still opaque. Our methodology employs the logits lens as a means to detect and analyze component information. However, this approach may encounter discrepancies between the middle layers and the output unembedding matrix. Such discrepancies can hinder a comprehensive and concrete analysis of the circuit components' behavior in the early layers. This limitation suggests the need for more robust techniques to bridge the gap between intermediate representations and final outputs. Recently, the Attention Lens method (Yuksekgonul et al., 2024) has been

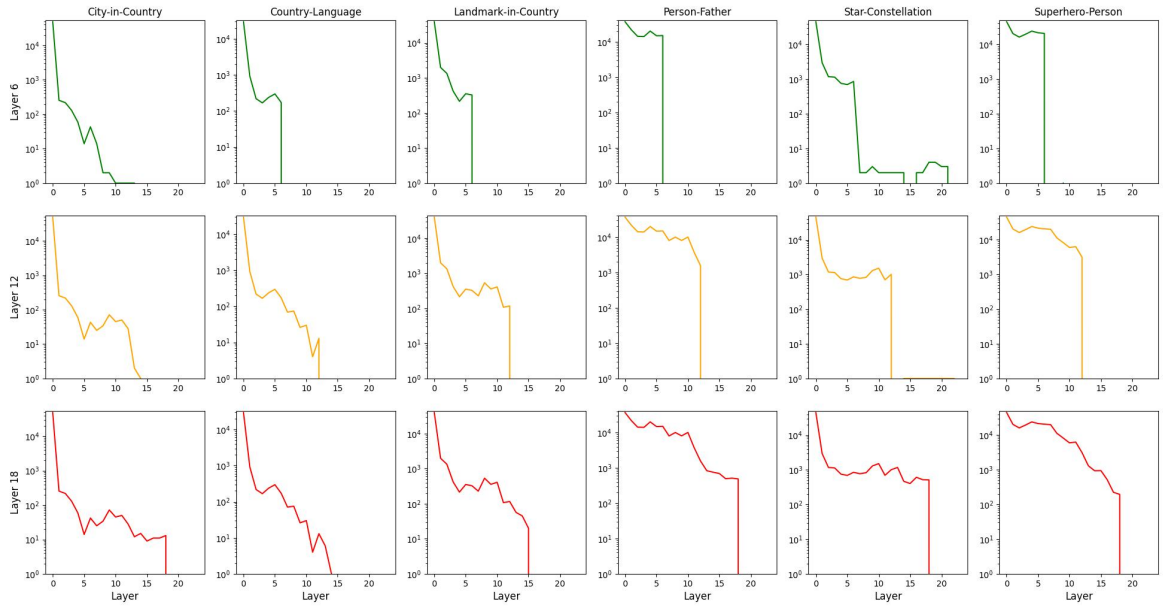


Figure 11: FT Rank Change Across Different Layers

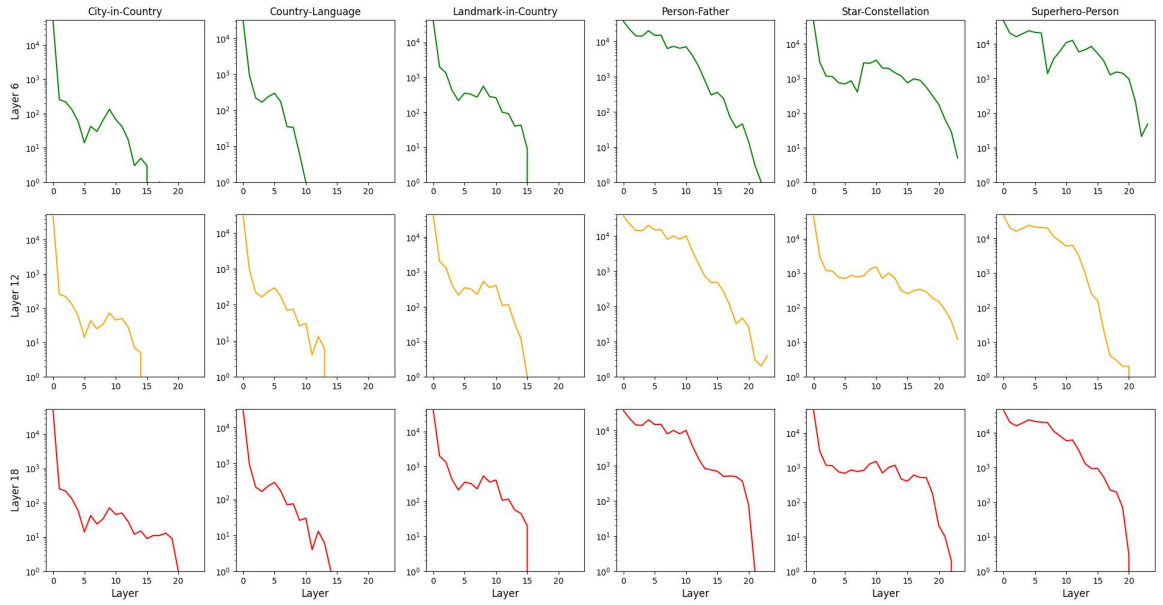


Figure 12: ROME Rank Change Across Different Layers

proposed, which involves training a specific unembedding matrix to map each attention head into the vocabulary space. While this method is promising, it is also resource-intensive. Nevertheless, it represents a potential starting point for a deeper understanding of the knowledge circuits within neural models. Moreover, our research indicates that several mover heads are reused across different types of knowledge or relational contexts. The mechanisms by which these heads are activated and the conditions under which they operate require further exploration and may shed light on why neurons are sometimes "monosemantic" responding to a single feature, and sometimes "polysemantic" (Elhage et al., 2022) responding to many unrelated features.