How do language models generalize to implications of facts they are trained on?

Jiahai Feng [1] **Stuart Russell** [1] **Jacob Steinhardt** [1]

Abstract

Pretrained language models (LMs) can generalize to implications of facts that they are finetuned on. For example, if finetuned on "John Doe lives in Tokyo", LMs can correctly answer "What language does the people in John Doe's city speak?" with "Japanese". We introduce the *extractive structures* framework for describing how different components in LMs (e.g. MLPs or attention heads) coordinate to enable this generalization. The structures consist of *informative components* that store training facts as weight changes, and *upstream* and *downstream extractive components* that query and process the stored information to produce the correct implication. We hypothesize that extractive structures are learned during pretraining when encountering implications of previously known facts. This yields two predictions: a data ordering effect where extractive structures can be learned only if facts precede their implications, and a weight grafting effect where extractive structures can be transferred without transferring the implications themselves. We validate our hypotheses in the OLMo-7B model on a synthetic two-hop reasoning setting. Of independent interest to the knowledge editing community, our results also indicate that fact learning can occur at an early and a late site, each of which enables different forms of generalization.

1. Introduction

Pretrained language models (LMs), when finetuned on a set of facts, can generalize to their implications. For example, if finetuned on "John Doe lives in Tokyo", LMs can correctly answer "What language does the people in John Doe's city speak?" with "Japanese". This generalization, dubbed "ripple effects" (Cohen et al., 2024) or "out of context reasoning" (OCR), occurs robustly for some implicatures (Berglund et al., 2023a; Treutlein et al., 2024), but is completely absent for others (e.g. the "reversal curse" phenomenon) (Berglund et al., 2023b; Allen-Zhu & Li, 2023).

Studying the OCR phenomenon could lead to a deeper theory of when and how LMs generalize from training data. The pretraining process is largely opaque, and practitioners are often surprised by emergent capabilities that LMs learn (Wei et al., 2022; Steinhardt, 2023). Such a theory could perhaps predict the existence or the impossibility of certain capabilities, and could be useful to building safe machine learning models (Bengio et al., 2023; Hubinger et al., 2019). Unfortunately, little is known about the origins and mechanisms of OCR.

We view the OCR phenomenon as communication between the backward pass on facts during finetuning and the forward pass on their implications at test time. Specifically, during finetuning the model receives a fact, which backpropagation has to encode as a set of weight changes to the model. Later at test time, the model is queried about the implication of the fact, and the forward pass on the query has to decode the correct answer from the new weights.

The communication viewpoint reveals two subproblems, which we develop conceptual and empirical tools to study. First, what are the encoding and decoding protocols? There must be mechanisms implemented in the weights of various components in the LM that encode facts at train time and decode them at inference time. Second, how do the encoder and decoder learn to coordinate with each other? OCR occurs in models pretrained with the standard self-supervised training objective, which must somehow lead to coordination between the finetune-time backward pass and the test-time forward pass.

To model the mechanisms underlying OCR, we propose the extractive structures framework, which posits that three groups of LM components (i.e. an attention head or MLP) work together at test time to enable OCR (Sec. **??**). Specifically, we posit that while during finetuning all components undergo weight changes, only a few *informative components* carry weight changes that are salient for predicting the implication (Fig. 1). Then, when testing on the implication, *upstream extractive components* process the input prompt into activations that elicit the information encoded in the weight change of informative components. Lastly, *downstream extractive components* decode the information

[1]UC Berkeley. Correspondence to: Jiahai Feng <fjiahai@berkeley.edu>.

Working draft.

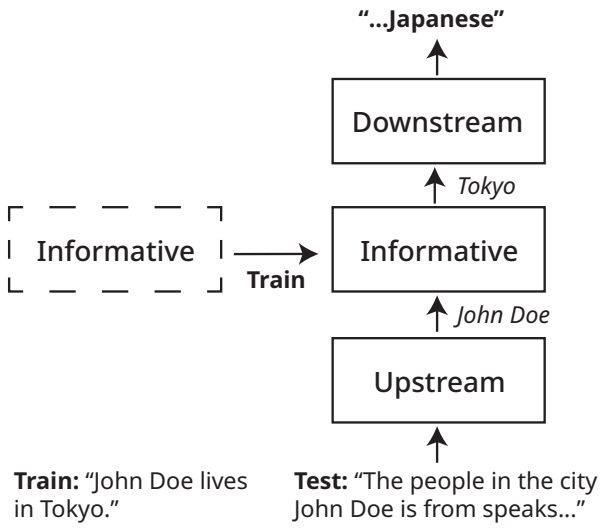


Figure 1. Extractive structures describe how LMs can generalize to facts they are trained on. Finetuning on the fact "John Doe lives in Tokyo" encodes the association "John Doe"→"Tokyo" in the weights of informative components. At test time, upstream structures elicits the fact from informative components by querying them with the right input ("John Doe"), whereas downstream structures post-processes the extracted information into the correct response ("Tokyo"→"Japanese").

produced by the informative components and post-process it to produce the correct implication.

We find empirically that in the OLMo language model (Groeneveld et al., 2024) on the two-hop reasoning task (Zhong et al., 2023), extractive structures can be localized to LM components, and are qualitatively consistent with prior mechanistic analyses of two-hop reasoning (Yang et al., 2024) (Sec. 5). To localize extractive structures we develop a set of metrics for deep linear networks (Sec. **??**) and extend them to language models (Sec. **??**). Further, our technique reveals that fact learning occurs in an early and a late site, each of which enables a different kind of generalization (Sec. 5.3). This is of independent interest to the knowledge editing community.

To explain how extractive structures arise, we hypothesize that the extractive components are learned during pretraining in the backward pass on implications of facts the model already knows. This solves the coordination problem: under this hypothesis, the backward pass on facts is free to encode facts according to an arbitrary schema, as long as later in training, extractive structures can learn to adapt to the encoding schema to decode facts.

We validate the hypothesis in a continued pretraining setting with new implicatures by testing two predictions. First, we observe a data ordering effect where the model fails at OCR if all facts occur after their implications during training (Sec. 6.1). Second, we observe that weight-space arithmetic can transfer newly learned extractive structures without also transferring the implications themselves (Sec. 6.2).

Overall, our work takes an empirical, mechanistic approach towards understanding how language models learn from their training data. We hope that our extractive structures framework can provide clarity for future work scoping out the capabilities and limitations of OCR.

2. Related works

Circuit-based Interpretability Circuit-based interpretability aims to decompose neural networks into components that form computational circuits (Cammarata et al., 2020; Elhage et al., 2021; Wang et al., 2022). These works often use causal techniques (Vig et al., 2020) to localize components (Meng et al., 2022), although gradient-based attribution scores analogous to Grad-CAM (Selvaraju et al., 2017; Olah et al., 2018) have seen a recent resurgence (Kramár et al., 2024; Grosse et al., 2023). Our work adapts these tools for analyzing components with weight changes.

Fact learning in LMs Fact learning in LMs and its robustness has been studied in the pretraining (Chang et al., 2024; Kandpal et al., 2023), finetuning (Berglund et al., 2023a;b), synthetic (Allen-Zhu & Li, 2023; Wang et al., 2024), and knowledge-editing (Cohen et al., 2024; Onoe et al., 2023) settings. While earlier works tended to analyze LM generalization behaviorally, recent works have used influence functions (Qin et al., 2024) and layer-wise ablations (Zhang et al., 2024) to characterize generalization. Our work provides a more fine-grained analysis by proposing concrete mechanisms for generalization.

Multi-hop reasoning Multi-hop reasoning is a common LM evaluation task where LMs have to compose several factual associations together (Zhong et al., 2023). In both pretrained LMs and LMs trained in grokking settings, researchers have found that LMs serially perform multi-hop reasoning by latently recalling intermediate hops (Yang et al., 2024; Wang et al., 2024). We use these results about trained LMs as a reference point when analyzing LMs' two-hop reasoning abilities as they learn new facts.

3. Background

In this section we concretely define OCR, and explain how two-hop reasoning can be an instance of OCR.

The core phenomenon we study is the observation that when a pretrained language model is finetuned on a set of facts $\mathcal{F}$, it can sometimes generalize to implications $\mathrm{Impl}\,\mathcal{F}$. Concretely, we model each fact $\mathsf{F} \in \mathcal{F}$ as the pair (p, a), where p is a prompt and $a \in \mathbb{A}_{\mathcal{F}}$ is a continuation of the

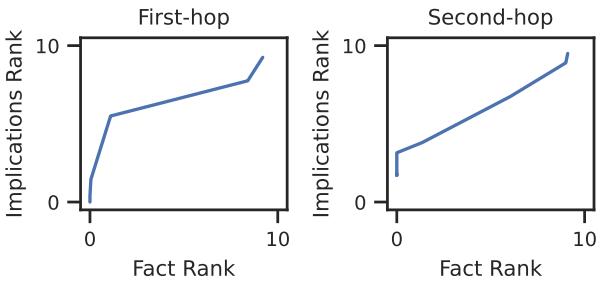


Figure 2. Mean ranks of facts and their implications throughout finetuning for first-hop and second-hop updates. Lower is better. Each point represents a training checkpoint, where the top right point is the initial pretrained LM. The mean ranks of facts and implications both fall over the course of 8 finetuning epochs, indicating that the LM generalizes to implications despite being only trained on facts.

First-hop update	
Fact *(Train)*	(John Doe lives in, Tokyo)
Impl. *(Test)*	(People in the city John Doe is from speak, Japanese)
Second-hop update	
Fact *(Train)*	(The mayor of Tokyo is, John Doe)
Impl. *(Test)*	(The mayor of the city that contains Senshoji temple is, John Doe)

Table 1. Illustrative examples from the first-hop update and second-hop update datasets.

prompt. Similarly, its implication $\mathrm{Impl}\, F$ is also a pair (p, a), $a \in \mathbb{A}_{\mathrm{Impl}\, \mathcal{F}}$. For example, a fact could be $\mathsf{F} =$ ("John Doe lives in", "Tokyo"), and $\mathrm{Impl}\, \mathsf{F} =$ ("The people from the city in which John Doe lives speak", "Japanese"). Then, the OCR phenomenon is when finetuning the model with initial weights $\mathbf{W}$ on the set $\mathcal{F}$ using standard cross-entropy loss leads to new weights $\mathbf{W}'$ that generalizes to implications $\mathrm{Impl}\, \mathcal{F}$.

To measure whether a model has generalized to an implication $\mathrm{Impl}\, \mathsf{F} = (p, a)$, we prompt the model with p, and measure the (0-indexed) rank of the continuation a out of all relevant continuations $\mathbb{A}_{\mathrm{Impl}\, \mathcal{F}}$. We use the mean rank when measuring generalization to a set of implications $\mathrm{Impl}\, \mathcal{F}$. Similarly, we use the mean rank for facts $\mathcal{F}$ to verify that the LM has indeed learned facts during finetuning.

The main form of implicatures we study relates to two-hop reasoning. When the LM is taught a novel fact, the LM may systematically compose it with existing facts to answer two-hop queries. For example, after teaching the model "John Doe lives in Tokyo", the model can compose this fact with existing knowledge that people in Tokyo speak Japanese to answer the prompt "The people from the city in which John Doe lives speak" with "Japanese".

There are two classes of implicatures with two-hop OCR, which composes a novel fact with a known fact. The novel fact could either be the first hop (first-hop update) or the second hop (second-hop update) of a two-hop reasoning chain. We construct synthetic datasets to study both (Sec. 5.1, Table 1). Fig. 2 shows that the LM is indeed capable of two-hop OCR in both settings.

4. Extractive structures framework

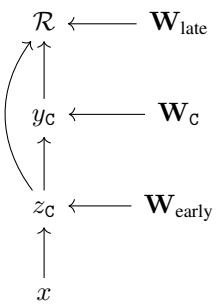
This section describes the extractive structures framework and proposes metrics for identifying components in the framework in language models.

Broadly speaking, the extractive structures framework views the ability of the model to know the implication $\mathrm{Impl}\, \mathsf{F} = (p, a)$ as a result of the coordination of three groups of components during the forward pass on the prompt p. Specifically, the extractive structures framework posits that the information about fact F is first saved in a group of *informative components* during the training process on the facts $\mathcal{F}$. Then, when prompted with an implication $\mathrm{Impl}\, \mathsf{F}$, a set of *upstream extractive structures* parses the input prompt p and provide the informative components a set of input activations that elicits the stored fact F from the informative component. Finally, a set of *downstream extractive structures* processes the recalled fact and ultimately produces the correct continuation a.

For each of the three groups of components, namely informative, upstream, and downstream components, we propose a metric based on a causal intervention on the computational graph of the language model. To make the metrics easier to compute, we linearly approximate causal metrics, and show that the three linearized metrics are first-order perturbations to a Grad-CAM style attribution score (Selvaraju et al., 2017).

Preliminaries Consider a feed-forward transformer that takes in inputs x and produces outputs y. For example, x could be the prompt p of an implication (p, a), and the outputs y could be a probability distribution over possible answers. Let the reward $\mathcal{R}$ be a real number that captures the extent to which the model correctly answered the prompt (e.g. log-probability of a).

Then, following conventions in interpretability literature, consider a particular component C in the LM, which could either an MLP or an attention head at a specific layer. Let z_{C} be its input, y_{C} its output, and $\mathbf{W}_{\mathsf{C}}$ be its weights so that $y_{\mathsf{C}} = f_{\mathsf{C}}(z_{\mathsf{C}}, \mathbf{W}_{\mathsf{C}})$. The inputs z_{C} to the component is an intermediate representation that is a function of the input x to the LM and the weights of the earlier layers $\mathbf{W}_{\text{early}}$, and similarly, the final reward $\mathcal{R}$ is dependent of the output of the component y_{C}, the weights of later layers $\mathbf{W}_{\text{late}}$, and, due to skip-connections in transformers, the component inputs z_{C}. The computational graph is summarized below.



Using this computational graph, we can express precisely the causal effects we expect each of the three groups of components to have. Specifically, we fix the input and desired output, and consider the reward $\mathcal{R}$, and how it changes under various interventions to the nodes in this graph. We denote an intervention that sets the value of a node v to a counterfactual value v' with $v \leftarrow v'$, and the resultant reward as $\mathcal{R}[v \leftarrow v']$. Let the original pretrained weights be denoted as the unprimed $\mathbf{W}$, and the finetuned weights be the primed $\mathbf{W}'$. The overall idea is that each of the three groups of components will allow the change in weights to increase rewards in different ways.

Informative components Intuitively, an informative component contains the newly learned knowledge that the rest of the network extracts. Therefore, if component $\mathtt{C}$ is an informative component, we expect setting $\mathbf{W}_{\mathtt{C}} \leftarrow \mathbf{W}'_{\mathtt{C}}$ while leaving all other weights unchanged to improve the reward. We thus write the informative score for component $\mathtt{C}$ as:

$$\mathcal{I}_{\mathtt{C}} = \mathcal{R}[\mathbf{W}_{\mathtt{C}} \leftarrow \mathbf{W}'_{\mathtt{C}}] - \mathcal{R} \tag{1}$$

Downstream components Intuitively, a downstream component processes the output containing the newly learned knowledge into useful output for the current prediction. Therefore, if component $\mathtt{C}$ is a downstream component, we expect its output $y_{\mathtt{C}}$ to help increase reward, but only if its input $z_{\mathtt{C}}$ contains the newly learned knowledge. We therefore compute the new $z'_{\mathtt{C}}$ with the new weights $\mathbf{W}'_{\text{early}}$ and contrast it with the original $z_{\mathtt{C}}$. Specifically, we compute the downstream score for component $\mathtt{C}$ as:

$$\mathcal{D}_{\mathtt{C}} = \mathcal{R}[y_{\mathtt{C}} \leftarrow f_{\mathtt{C}}(z'_{\mathtt{C}}, \mathbf{W}_{\mathtt{C}})] - \mathcal{R} \tag{2}$$

Upstream components Intuitively, an upstream component should process the input into an intermediate representation that engages the newly updated weights so as to retrieve the new information. Therefore, if component $\mathtt{C}$ is an upstream component, we expect its outputs to feed into later layers to increase reward, but only if the later layers contain the updated weights. We describe a way to measure the importance of component $\mathtt{C}$ for later layers, and then identify whether this importance measure increases when later layers have new weights.

A standard way of measuring importance of a node is to ablate its value by setting it to a constant, a common choice being the node's mean value over some reference set of inputs (Chan et al., 2022). Therefore, to evaluate the importance of component $\mathtt{C}$ we set its output $y_{\mathtt{C}}$ to some constant value $y^*_{\mathtt{C}}$, and measure the change in reward $\mathcal{R} - \mathcal{R}[y_{\mathtt{C}} \leftarrow y^*_{\mathtt{C}}]$.

We expect upstream components to be comparatively more important when future layers contain weights that depend on the outputs of upstream components. We therefore compare the importance of component $\mathtt{C}$ when future layers have updated weights $\mathbf{W}'_{\text{late}}$ to when they have the original weights $\mathbf{W}_{\text{late}}$. Specifically, the upstream score for component $\mathtt{C}$ is:

$$\mathcal{U}_{\mathtt{C}} = (\mathcal{R}[\mathbf{W}_{\text{late}} \leftarrow \mathbf{W}'_{\text{late}}] - \mathcal{R}[\mathbf{W}_{\text{late}} \leftarrow \mathbf{W}'_{\text{late}}, y_{\mathtt{C}} \leftarrow y^*_{\mathtt{C}}]) - (\mathcal{R} - \mathcal{R}[y_{\mathtt{C}} \leftarrow y^*_{\mathtt{C}}]) \tag{3}$$

Linearizing the extractive scores Computing the causal scores in language models require several passes over the model, one for each component. One common approximation (Selvaraju et al., 2017; Kramár et al., 2024) is to replace the causal interventions with their linear approximation. Specifically, for any node v in the computational graph, we approximate

$$\mathcal{R}[v \leftarrow v_1] - \mathcal{R}[v \leftarrow v_2] \approx \frac{\mathrm{d}\mathcal{R}}{\mathrm{d}v}(v_1 - v_2),$$

where the derivative can be evaluated at either v_1 or v_2. We thus obtain[1]

$$\overline{\mathcal{I}}_{\mathtt{C}} = \frac{\mathrm{d}\mathcal{R}}{\mathrm{d}\mathbf{W}_{\mathtt{C}}}(\mathbf{W}'_{\mathtt{C}} - \mathbf{W}_{\mathtt{C}}) \tag{4}$$

$$\overline{\mathcal{D}}_{\mathtt{C}} = \frac{\mathrm{d}\mathcal{R}}{\mathrm{d}y_{\mathtt{C}}}(f_{\mathtt{C}}(z'_{\mathtt{C}}, \mathbf{W}_{\mathtt{C}}) - y_{\mathtt{C}}) \tag{5}$$

$$\overline{\mathcal{U}}_{\mathtt{C}} = (\frac{\mathrm{d}\mathcal{R}}{\mathrm{d}y_{\mathtt{C}}}[\mathbf{W}_{\text{late}} \leftarrow \mathbf{W}'_{\text{late}}] - \frac{\mathrm{d}\mathcal{R}}{\mathrm{d}y_{\mathtt{C}}})(y_{\mathtt{C}} - y^*_{\mathtt{C}}). \tag{6}$$

The three linearized scores can be seen as first order perturbations to a single value. Consider the quantity

$$\overline{\mathcal{R}}_{\mathtt{C}} = \frac{\mathrm{d}\mathcal{R}}{\mathrm{d}y_{\mathtt{C}}}(f(z_{\mathtt{C}}, \mathbf{W}_{\mathtt{C}}) - y^*_{\mathtt{C}}),$$

which can be interpreted as an approximation to the effects of ablating the outputs of component $\mathtt{C}$ to the constant $y^*_{\mathtt{C}}$. Let $\breve{y} = \frac{\mathrm{d}\mathcal{R}}{\mathrm{d}y_{\mathtt{C}}}$ and $\breve{y}' = \frac{\mathrm{d}\mathcal{R}}{\mathrm{d}y_{\mathtt{C}}}[\mathbf{W}_{\text{late}} \leftarrow \mathbf{W}'_{\text{late}}]$. Then, with the chain rule, one can rewrite

$$\overline{\mathcal{I}}_{\mathtt{C}} = \breve{y}_{\mathtt{C}} \nabla_w f_{\mathtt{C}}(z_{\mathtt{C}}, \mathbf{W}_{\mathtt{C}})(\mathbf{W}'_{\mathtt{C}} - \mathbf{W}_{\mathtt{C}}) \tag{7}$$

$$\overline{\mathcal{D}}_{\mathtt{C}} = \breve{y}_{\mathtt{C}} \nabla_z f_{\mathtt{C}}(z_{\mathtt{C}}, \mathbf{W}_{\mathtt{C}})(z'_{\mathtt{C}} - z_{\mathtt{C}}) \tag{8}$$

$$\overline{\mathcal{U}}_{\mathtt{C}} = (\breve{y}'_{\mathtt{C}} - \breve{y}_{\mathtt{C}}) f_{\mathtt{C}}(z_{\mathtt{C}}, \mathbf{W}_{\mathtt{C}}) \tag{9}$$

[1]The linearized informative score is related to component-wise influence functions (Grosse et al., 2023).

Finally, notice that the difference between the primed and unprimed values $\delta\breve{y}$, $\delta\mathbf{W}_{\mathtt{C}}$, and $\delta z_{\mathtt{C}}$ correspond to changes in the late weights $\mathbf{W}_{\text{late}}$, component weights $\mathbf{W}_{\mathtt{C}}$, and early weights $\mathbf{W}_{\text{early}}$ respectively. We can thus write the change in reward $\delta\overline{\mathcal{R}}_{\mathtt{C}}$ due to the overall change in weights $\delta\mathbf{W}$ in terms of first-order perturbations in each of the three sets of weights, so that with the chain and product rules:

$$\delta\overline{\mathcal{R}}_{\mathtt{C}} = \underbrace{\delta\breve{y}_{\mathtt{C}}(f_{\mathtt{C}}(\mathbf{W}_{\mathtt{C}}, z_{\mathtt{C}}) - y^*_{\mathtt{C}})}_{\text{Upstream score } \overline{\mathcal{U}}_{\mathtt{C}}} + \underbrace{\breve{y}_{\mathtt{C}}(\nabla_{\mathbf{W}_{\mathtt{C}}} f_{\mathtt{C}}(\mathbf{W}_{\mathtt{C}}, z_{\mathtt{C}})\,\delta\mathbf{W}_{\mathtt{C}})}_{\text{Informational score } \overline{\mathcal{I}}_{\mathtt{C}}} + \underbrace{\breve{y}_{\mathtt{C}}(\nabla_{z_{\mathtt{C}}} f_{\mathtt{C}}(\mathbf{W}_{\mathtt{C}}, z_{\mathtt{C}})\,\delta z_{\mathtt{C}})}_{\text{Downstream score } \overline{\mathcal{D}}_{\mathtt{C}}}$$

This interprets the three scores as first-order changes to a gradient-based attribution score, and suggests that the three scores ought to be comparable in magnitude.

5. Extractive structures in two-hop reasoning

In this section, we compute the three extractive scores for a language model on first-hop and second-hop updates, and argue that the extractive scores are consistent with prior mechanistic analysis of two-hop reasoning. We then causally verify the informative scores, and along the way we find that fact learning occurs at an early site and a late site, so that the early site enables first-hop updates and the late site enables second-hop updates. This last finding may be independently interesting to the knowledge editing community.

5.1. Setup

Dataset We create two synthetic datasets to study OCR in two-hop reasoning. We gather a set of 20 cities, together with an associated language and an associated landmark. We then gather set of 20 fictitious names, which we pair randomly with the 20 cities. In first-hop dataset, facts are associations from names to cities, and implications are queries about the language associated a person's city. In the second-hop dataset, facts are associations from city to names, and implications are queries about the person associated with the city a landmark is in. We use a simple template to populate facts and implications (see Table 1 for examples).

Model We use the OLMo 7B model as the base model. We use the intermediate training checkpoint right before the final annealing phase to simulate a continued pretraining setting where the language model is encountering new facts in its pretraining.

Training We finetune the model on the set of facts $\mathcal{F}$ using the standard cross-entropy loss. We only include the loss on answer tokens. We use the Adam optimizer (Kingma & Ba, 2014) for 8 epochs at 3×10^{-6} learning rate, momentum $(0.9, 0.999)$, batch size 8, with a linear rate decay schedule with 10% warmup.

5.2. Results

We finetune the model on the first-hop and second-hop facts, and compute the three extractive scores on their respective implications (Fig. 3). The most salient observation is that the informative components on MLPs are mostly localized to the early/middle layers of the name tokens in first-hop updates (row 2, column 1), whereas for second-hop they are mostly localized to the late layers at the last token (row 2, column 3). The downstream scores are mostly in the late layers of the last token, although there is some presence in the middle layers of the name tokens. The upstream scores are localized to the early layers in first-hop update, but are instead mostly found in the late layers in the last token for second-hop updates, right before the informative components.

The extractive scores are consistent with prior mechanistic analyses of two-hop reasoning. Yang et al. (2024) found evidence that, to answer a two-hop query, language models latently look up the first fact, and then use the result to look up the second fact. In our setting, both the downstream components of first-hop updates and the informative components of second-hop updates can be interpreted as the components responsible for recalling the second hop, and they both point to the last layers at the last token. Similarly, both the upstream components of second-hop updates and the informative components of the first-hop updates can be interpreted as components responsible for recalling the first hop, and they both point to the early-middle layers of the landmark/name tokens.

The extractive scores also have suggestive features that are not yet well understood in mechanistic interpretability. Prior mechanistic interpretability works on factual recall (Geva et al., 2023; 2020; Meng et al., 2022) tend to regard MLPs as associative memories that recall factual information from their weights, whereas attention heads transfer information from one token position to another. Consistent with this view, we find that for first-hop updates, the early-middle MLP layers are the name tokens are informative. However, in addition to these MLP components, the attention heads at the late layers on the last token are also informative, suggesting that these attention heads may also play a role in recalling information.

5.3. Causal analysis

We verify the earlier extractive scores with causal experiments, and in so doing identify that fact learning occurs at an early and a late site. Specifically, the two-hop informative scores (Fig. 3) suggest that to perform first-hop updates, the

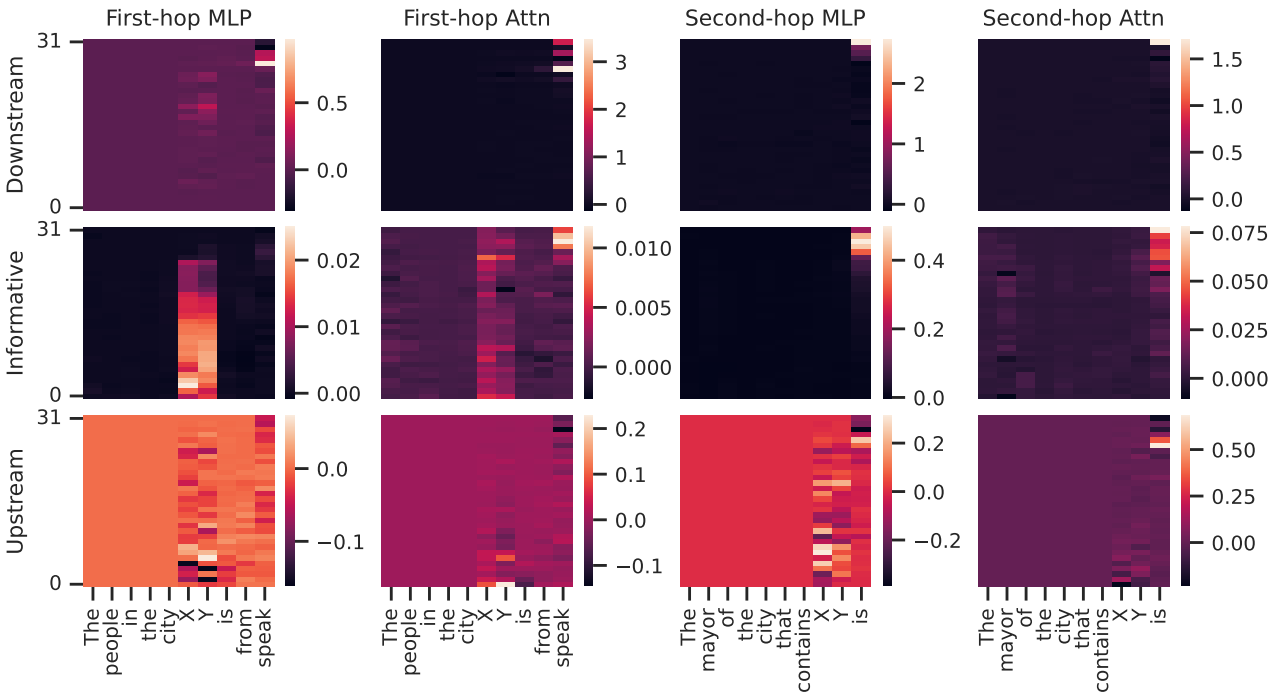


Figure 3. Extractive scores for the first-hop (left) and second-hop updates (right). Scores are averaged over the dataset.

	First-hop		Second-hop	
Frozen Layers	Fact	Impl.	Fact	Impl.
None	0.00	0.00	0.00	1.80
Early	5.10	8.40	0.95	1.85
Late	0.00	0.00	0.10	6.30
All	9.20	9.25	9.10	9.50
Early (pre)	0.00	6.50	0.00	0.50
Late (pre)	0.00	0.10	0.00	6.60

Table 2. Mean-rank of facts and implications when freezing weights post-training or pre-training. Freezing early layers harm first-hop OCR but not second-hop OCR, and vice versa for late layers. 'None' and 'All' are baselines where we use the full fine-tuned weights and original weights respectively.

facts should be encoded in the early-middle layers (first 2/3), whereas to perform second-hop updates, the facts should be encoded in the late layers (last 1/3).

We test this causally by freezing certain layers of weights and checking if the model retains its knowledge of facts and implications. Specifically, we finetune the model to obtain a weight change as before, but then freeze a subset of components (either early-middle or late) to their original weights. This is a direct test of the causal faithfulness of the informative score. As a secondary test, we instead freeze the subset of components before finetuning, and re-finetune the model with restricted set of free parameters. This checks that the extractive structures are indeed localized to the components identified by the extractive scores, and do not vary based on where the training facts are updated.

Our findings confirm that freezing the early-middle layers harms the first-hop implications but not second-hop implications, and vice versa for the late layers (Table 2). Specifically, freezing the early-middle layers increases the mean-rank of first-hop implications from 0.00 to 8.40, but the second-hop implications from 1.80 to 1.85, and freezing late layers leaves the first-hop implications unchanged at 0.00, whereas the second-hop implications increased from 1.80 to 6.30. This validates that the informative score correctly captures where the salient information is stored.

In the setting where the weights are frozen before finetuning, we see that the model adapts to instead store facts at different layers, but this is insufficient to recover OCR. If the early-middle layers are frozen post-finetuning, the model forgets a lot of first-hop facts (5.10 mean rank), but if they are frozen pre-finetuning, the model has a chance to instead store the facts at the late layers, and retains the first-hop facts (0.00 mean rank). However, this adaptation is insufficient to also recover first-hop implications (6.50 mean rank), because the downstream components that depend on the new first-hop facts are unable to utilize the first-hop facts stored late in the model. In contrast, freezing early-middle layers still allow second-hop implications to be learned (0.50 mean rank). Conversely, freezing late layers pre-finetuning still learns first-hop implications (0.10 mean rank) but not second-op implications (6.60 mean rank).

Our results suggest that while fact learning stores facts at

many points in the LM, storing facts at different points in the network enable different forms of generalization. Early knowledge editing works (Meng et al., 2022) suggested that facts are stored in a localized set of components, but later work (Hase et al., 2023) found that facts can be edited at any point in the LM. Our results suggests a more nuanced picture: facts can indeed be stored at any point in the network (since "Early (pre)" and "Late (pre)" both learn facts well); however, storing facts in the early layers enable first-hop updates, whereas the late layers enable second-hop updates.

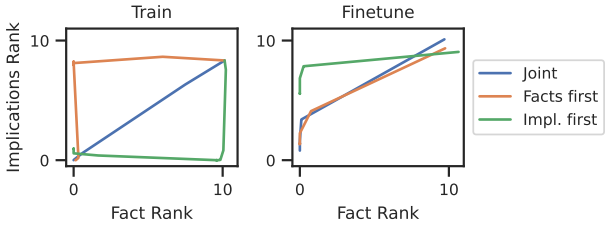


Figure 4. Mean ranks for facts and their implicatures while continued pretraining on training facts $\mathcal{F}_{\text{train}}$ (left), and while subsequently finetuning on test facts $\mathcal{F}_{\text{test}}$ (right).

6. Origins of extractive structures

We now turn our attention to the coordination problem between the backward pass on the train fact F and the forward pass on the implication $\text{Impl}\,\text{F}$. Is it just a coincidence that the backward pass modifies the weights in such a way that extractive components can exploit them?

We hypothesize that the coordination problem is solved by the backward passes on implications during training reinforcing extractive structures. Specifically, suppose the model already knows a fact F and then later encounters its implication $\text{Impl}\,F$; this could happen by chance from training data shuffling. Then, training on the implication creates training signals for the formation of extractive structures that seek out the known fact F and returns $\text{Impl}\,F$. This solves the coordination problem because the initial learning of facts has relatively large freedom to encode the information, since the subsequent backward pass on implications can adapt to the choice of information encoding.

This hypothesis has two testable implications: a data ordering effect (Sec. 6.1) and the ability to graft weight changes corresponding to new extractive structures without also grafting the implications themselves (Sec. 6.2).

6.1. Data ordering

The hypothesis predicts a data ordering effect during pretraining, where if all the facts appear after their implications, then the model cannot learn extractive structures, and hence cannot later generalize to implications of new facts. Conversely, if all facts precede their implications, then extractive structures may form and later enable OCR.

At a high level, we study the how pretraining learns implicatures by creating a dataset with novel implicatures, and studying how finetuning a pretrained model on this dataset creates new extractive structures for the new implicatures. This finetuning process simulates what would have happened if the pretraining data had contained these novel implicatures, and for clarity we call this phase *continued pretraining*. Then, if continued pretraining successfully created new extractive structures, we expect finetuning the new model on new facts would cause it to generalize to their implications according to the newly learned implicatures.

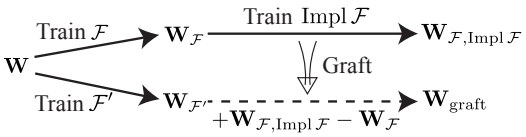


Figure 5. Visualization of grafting procedure

Setup We create a dataset of novel implicatures. Different from the earlier setting where the associations underlying implicatures are already known (e.g. "Tokyo" → "Japanese"), we aim to teach completely new implicatures involving new relations. Specifically, we create the "dax" and "wug" relations that associate a person with a city or an animal respectively. We then assign each city a random animal (e.g. "Tokyo" → "tiger") so that if "John Doe dax Tokyo" is true, then so is "John Doe wug the tiger". Concretely, we construct a set of 80 facts $\mathcal{F}_{\text{train}}$ by randomly assigning to each of 80 names one of 20 cities, and a set of 80 corresponding implications by using the assigned city's animal. To evaluate OCR ability, we additionally create a test dataset of 20 facts $\mathcal{F}_{\text{test}}$ and their implications by pairing 20 new names with the 20 cities/animals.

We continually pretrain the model on the training facts and their implications $\mathcal{F}_{\text{train}} \cup \text{Impl}\,\mathcal{F}_{\text{train}}$ and test if data ordering affects its ability to perform OCR on the test facts $\mathcal{F}_{\text{test}}$. Specifically, we investigate three orderings of the training data: facts-then-implications, implications-then-facts, and where facts and implications are shuffled together. To evaluate OCR, we finetune the model on test facts $\mathcal{F}_{\text{test}}$, and measure its generalization to the test implications $\text{Impl}\,\mathcal{F}_{\text{test}}$.

Results Fig. 4 (right) confirms the prediction that the model can perform OCR only in the data orderings where facts precede implications and where they are mixed, but not where implications precede facts. Importantly, this happens despite the fact that the model successfully learns the training facts and implications in all three data orderings (Fig. 4 left). This highlights that the internal extractive structures can affect OCR generalization ability in ways that are not captured by conventional metrics.

Weights	Impl $\mathcal{F}'$	Impl $\mathcal{F}$	$\mathcal{F}'$
$\mathbf{W}_{\mathcal{F}'}$	9.13	8.55	0.00
$\mathbf{W}_{\text{graft}}$	1.13	3.30	0.10
$\widetilde{\mathbf{W}}_{\text{graft}}$	8.38	0.43	0.32

Table 3. Mean ranks for facts and their implicatures while training on training facts (left), and while subsequently training on test facts (right).

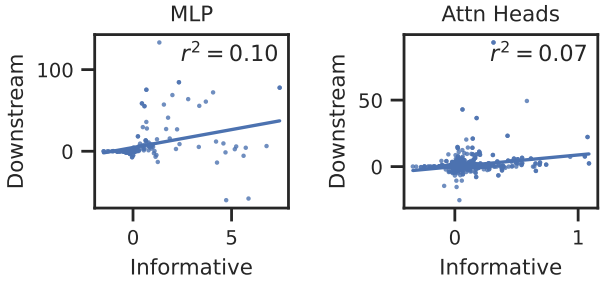


Figure 6. Correlation between downstream extractive scores and informative scores for MLPs (left) and attention heads (right).

6.2. Weight grafting

Because extractive structures are localized, we might expect weight-space arithmetic to transfer newly formed extractive components without transferring the underlying facts. Specifically, suppose we first teach the model facts $\mathcal{F}$, to produce weights $\mathbf{W}_{\mathcal{F}}$, and then teach the model extractive components by training on implications $\text{Impl}\,\mathcal{F}$ to produce weights $\mathbf{W}_{\mathcal{F},\text{Impl}\,\mathcal{F}}$ (Fig. 5). Then, the difference in weights $\mathbf{W}_{\mathcal{F},\text{Impl}\,\mathcal{F}} - \mathbf{W}_{\mathcal{F}}$ ought to be the weight changes needed to create extractive components, and should be localized to the extractive components. Therefore, if we instead train the model on a counterfactual set of facts $\mathcal{F}'$ to produce $\mathbf{W}_{\mathcal{F}'}$, and graft the weight difference over to produce the weights $\mathbf{W}_{\text{graft}} = \mathbf{W}_{\mathcal{F}'} + \mathbf{W}_{\mathcal{F},\text{Impl}\,\mathcal{F}} - \mathbf{W}_{\mathcal{F}}$, we should have a model that has learned the counterfactual implications $\text{Impl}\,\mathcal{F}'$ instead of the original implications $\text{Impl}\,\mathcal{F}$, because we should have grafted over the extractive components and not the original implications themselves. Conversely, if instead of grafting over the extractive components $\mathbf{W}_{\mathcal{F},\text{Impl}\,\mathcal{F}} - \mathbf{W}_{\mathcal{F}}$, we graft over the implications $\mathbf{W}_{\text{Impl}\,\mathcal{F}} - \mathbf{W}$ to produce $\widetilde{\mathbf{W}}_{\text{graft}} = \mathbf{W}_{\mathcal{F}'} + \mathbf{W}_{\text{Impl}\,\mathcal{F}} - \mathbf{W}$, we expect to have transferred the implications $\text{Impl}\,\mathcal{F}$ but not the extractive components for producing $\text{Impl}\,\mathcal{F}'$.

We further verify that the grafted weights do indeed contain the extractive components by measuring their extractive scores in two different manners and correlating them. First, suppose the grafted weights contain downstream components. Then, we expect the same downstream components to be active in helping $\mathbf{W}_{\mathcal{F},\text{Impl}\,\mathcal{F}}$ perform OCR when fine-tuned on test facts, and be identifiable with the downstream score. The second approach considers the grafting update $\mathbf{W}_{\mathcal{F}'} \to \mathbf{W}_{\text{graft}}$ as an update that teaches the model the implications $\text{Impl}\,\mathcal{F}'$. We therefore expect the downstream components to be identifiable with the informative score of this update.

Setup We use the same train and test datasets as the data ordering experiment (Sec. 6.1), but additionally create a counterfactual version of the train dataset by re-assigning random cities/animals to the 80 names.

Results Table 3 confirms the weight grafting hypothesis that extractive components for producing implications can be transferrred without the implications themselves. We observe that grafting over the weight difference $\mathbf{W}_{\mathcal{F},\text{Impl}\,\mathcal{F}} - \mathbf{W}_{\mathcal{F}}$ has the main effect of lowering the mean rank for counterfactual implications $\text{Impl}\,\mathcal{F}'$ from 9.13 to 1.13, while secondarily lowering the mean rank for the original implications $\text{Impl}\,\mathcal{F}$ by a smaller extent to 3.30. In contrast, grafting over $\mathbf{W}_{\text{Impl}\,\mathcal{F}} - \mathbf{W}$ has the main effect of transferring implications $\text{Impl}\,\mathcal{F}$ ($8.55 \to 0.43$), but not the extractive structures for deducing implications $\text{Impl}\,\mathcal{F}'$ ($9.13 \to 8.38$). However, we note that while the grafted weights $\mathbf{W}_{\text{graft}}$ primarily contain the extractive structures for $\text{Impl}\,F'$, they do nonetheless contain some memorized original implications $\text{Impl}\,F$, suggesting that when trained on implications, models both memorize the implications and learn extractive structures that extract them from known facts.

To further confirm that the grafted weights contain the downstream components, we correlate the two approaches for localizing them (Fig. 6). We find that for both MLP components and attention heads, there is a statistically significant positive correlation between the two metrics. However, the Pearson correlation is small, suggesting that the two metrics do not align perfectly.

7. Conclusion

This work studies the empirical phenomenon that pretrained language models can some times generalize to implications of facts they are trained on. To this end, we develop the extractive structures framework for explaining how the forward pass on implications can extract and process facts stored during training. Additionally, we find evidence that extractive components are learned during pretraining, where training on implications of already known facts creates a learning signal to form extractive components.

We hope that our work will enable further analyses of empirical deep learning phenomena, which could lead to a general theory of deep learning generalization. For example, identifying sufficient conditions for extractive structures or classifying all possible extractive structures could provide strong bounds on what forms of generalizations are possible.

This would be an ambitious result of relevance to understanding the success of deep learning and to characterize their emergent behaviors.

Impact Statement

This paper aims to deepen our understanding of empirical deep learning phenomena. We believe such understanding may potentially lead to stronger theories of deep learning generalization, which could help society develop and deploy deep learning systems more safely.

References

Allen-Zhu, Z. and Li, Y. Physics of language models: Part 3.2, knowledge manipulation. *arXiv preprint arXiv:2309.14402*, 2023.

Bengio, Y., Hinton, G., Yao, A., Song, D., Abbeel, P., Harari, Y. N., Zhang, Y.-Q., Xue, L., Shalev-Shwartz, S., Hadfield, G., et al. Managing ai risks in an era of rapid progress. *arXiv preprint arXiv:2310.17688*, 2023.

Berglund, L., Stickland, A. C., Balesni, M., Kaufmann, M., Tong, M., Korbak, T., Kokotajlo, D., and Evans, O. Taken out of context: On measuring situational awareness in llms. *arXiv preprint arXiv:2309.00667*, 2023a.

Berglund, L., Tong, M., Kaufmann, M., Balesni, M., Stickland, A. C., Korbak, T., and Evans, O. The reversal curse: Llms trained on" a is b" fail to learn" b is a". *arXiv preprint arXiv:2309.12288*, 2023b.

Cammarata, N., Carter, S., Goh, G., Olah, C., Petrov, M., Schubert, L., Voss, C., Egan, B., and Lim, S. K. Thread: circuits. *Distill*, 5(3):e24, 2020.

Chan, L., Garriga-Alonso, A., Goldowsky-Dill, N., Greenblatt, R., Nitishinskaya, J., Radhakrishnan, A., Shlegeris, B., and Thomas, N. Causal scrubbing: A method for rigorously testing interpretability hypotheses. In *AI Alignment Forum*, pp. 10, 2022.

Chang, H., Park, J., Ye, S., Yang, S., Seo, Y., Chang, D.-S., and Seo, M. How do large language models acquire factual knowledge during pretraining? *arXiv preprint arXiv:2406.11813*, 2024.

Cohen, R., Biran, E., Yoran, O., Globerson, A., and Geva, M. Evaluating the ripple effects of knowledge editing in language models. *Transactions of the Association for Computational Linguistics*, 12:283–298, 2024.

Elhage, N., Nanda, N., Olsson, C., Henighan, T., Joseph, N., Mann, B., Askell, A., Bai, Y., Chen, A., Conerly, T., DasSarma, N., Drain, D., Ganguli, D., Hatfield-Dodds, Z., Hernandez, D., Jones, A., Kernion, J., Lovitt, L., Ndousse, K., Amodei, D., Brown, T., Clark, J., Kaplan, J., McCandlish, S., and Olah, C. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 2021. https://transformer-circuits.pub/2021/framework/index.html.

Geva, M., Schuster, R., Berant, J., and Levy, O. Transformer feed-forward layers are key-value memories. *ArXiv*, abs/2012.14913, 2020. URL https://api.semanticscholar.org/CorpusID:229923720.

Geva, M., Bastings, J., Filippova, K., and Globerson, A. Dissecting recall of factual associations in auto-regressive language models. *arXiv preprint arXiv:2304.14767*, 2023.

Groeneveld, D., Beltagy, I., Walsh, P., Bhagia, A., Kinney, R., Tafjord, O., Jha, A. H., Ivison, H., Magnusson, I., Wang, Y., et al. Olmo: Accelerating the science of language models. *arXiv preprint arXiv:2402.00838*, 2024.

Grosse, R., Bae, J., Anil, C., Elhage, N., Tamkin, A., Tajdini, A., Steiner, B., Li, D., Durmus, E., Perez, E., et al. Studying large language model generalization with influence functions. *arXiv preprint arXiv:2308.03296*, 2023.

Hase, P., Bansal, M., Kim, B., and Ghandeharioun, A. Does localization inform editing? surprising differences in causality-based localization vs. *Knowledge Editing in Language Models*, 2023.

Hubinger, E., van Merwijk, C., Mikulik, V., Skalse, J., and Garrabrant, S. Risks from learned optimization in advanced machine learning systems. *arXiv preprint arXiv:1906.01820*, 2019.

Kandpal, N., Deng, H., Roberts, A., Wallace, E., and Raffel, C. Large language models struggle to learn long-tail knowledge. In *International Conference on Machine Learning*, pp. 15696–15707. PMLR, 2023.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kramár, J., Lieberum, T., Shah, R., and Nanda, N. Atp*: An efficient and scalable method for localizing llm behaviour to components. *arXiv preprint arXiv:2403.00745*, 2024.

Meng, K., Bau, D., Andonian, A., and Belinkov, Y. Locating and editing factual associations in gpt. *Advances in Neural Information Processing Systems*, 35:17359–17372, 2022.

Olah, C., Satyanarayan, A., Johnson, I., Carter, S., Schubert, L., Ye, K., and Mordvintsev, A. The building blocks of interpretability. *Distill*, 3(3):e10, 2018.

Onoe, Y., Zhang, M. J., Padmanabhan, S., Durrett, G., and Choi, E. Can lms learn new entities from descriptions? challenges in propagating injected knowledge. *arXiv preprint arXiv:2305.01651*, 2023.

Qin, J., Zhang, Z., Han, C., Li, M., Yu, P., and Ji, H. Why does new knowledge create messy ripple effects in llms? *arXiv preprint arXiv:2407.12828*, 2024.

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., and Batra, D. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pp. 618–626, 2017.

Steinhardt, J. Emergent deception and emergent optimization. https://web.archive.org/web/20240324025422/https://bounded-regret.ghost.io/emergent-deception-optimization/, 2023.

Treutlein, J., Choi, D., Betley, J., Anil, C., Marks, S., Grosse, R. B., and Evans, O. Connecting the dots: Llms can infer and verbalize latent structure from disparate training data. *arXiv preprint arXiv:2406.14546*, 2024.

Vig, J., Gehrmann, S., Belinkov, Y., Qian, S., Nevo, D., Singer, Y., and Shieber, S. Investigating gender bias in language models using causal mediation analysis. *Advances in neural information processing systems*, 33: 12388–12401, 2020.

Wang, B., Yue, X., Su, Y., and Sun, H. Grokked transformers are implicit reasoners: A mechanistic journey to the edge of generalization. *arXiv preprint arXiv:2405.15071*, 2024.

Wang, K., Variengien, A., Conmy, A., Shlegeris, B., and Steinhardt, J. Interpretability in the wild: A circuit for indirect object identification in gpt-2 small. *arXiv preprint arXiv:2211.00593*, 2022.

Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., Yogatama, D., Bosma, M., Zhou, D., Metzler, D., et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682*, 2022.

Yang, S., Gribovskaya, E., Kassner, N., Geva, M., and Riedel, S. Do large language models latently perform multi-hop reasoning? *arXiv preprint arXiv:2402.16837*, 2024.

Zhang, X., Li, M., and Wu, J. Co-occurrence is not factual association in language models. *arXiv preprint arXiv:2409.14057*, 2024.

Zhong, Z., Wu, Z., Manning, C. D., Potts, C., and Chen, D. Mquake: Assessing knowledge editing in language models via multi-hop questions. *arXiv preprint arXiv:2305.14795*, 2023.