



Out-of-distribution generalization via composition: A lens through induction heads in Transformers

Jiajun Song^a , Zhuoyan Xu^b, and Yiqiao Zhong^{b,1}

Edited by David Weitz, Harvard University, Cambridge, MA; received August 23, 2024; accepted January 5, 2025

Large language models (LLMs) such as GPT-4 sometimes appear to be creative, solving novel tasks often with a few demonstrations in the prompt. These tasks require the models to generalize on distributions different from those from training data—which is known as out-of-distribution (OOD) generalization. Despite the tremendous success of LLMs, how they approach OOD generalization remains an open and underexplored question. We examine OOD generalization in settings where instances are generated according to hidden rules, including in-context learning with symbolic reasoning. Models are required to infer the hidden rules behind input prompts without any fine-tuning. We empirically examined the training dynamics of Transformers on a synthetic example and conducted extensive experiments on a variety of pretrained LLMs, focusing on a type of component known as induction heads. We found that OOD generalization and composition are tied together—models can learn rules by composing two self-attention layers, thereby achieving OOD generalization. Furthermore, a shared latent subspace in the embedding (or feature) space acts as a bridge for composition by aligning early layers and later layers, which we refer to as the common bridge representation hypothesis.

out-of-distribution generalization | composition | large language models | emergent ability | mechanistic interpretability

Large language models (LLMs) are sometimes able to solve complex tasks that appear novel or require reasoning abilities. The appearance of creativity in task-solving has sparked recent discussions about artificial general intelligence (1–4).

The classical notion of statistical generalization does not fully explain the progress observed in LLMs. Traditionally, both training instances and test instances are drawn from the same distribution, and it is generally not expected that a model will generalize well on a different test distribution without explicit domain adaptation involving model updates.

The recent success of LLMs suggests a different story: if test data involve compositional structures, LLMs can generalize across different distributions with just a few demonstrations in the prompt (few-shot learning) or even without any demonstrations (zero-shot learning) (5) without updating the model parameters. Indeed, the apparent ability of models to infer rules from the context of a prompt—known as in-context learning (ICL)—is a hallmark of LLMs (6). Moreover, a growing body of literature on chain-of-thought prompting explicitly exploits the compositional structures of reasoning tasks. For example, phrases like “let’s think step by step” are prepended to input prompts to elicit reasoning (7, 8), and prompts with intermediate steps are used to improve accuracy on mathematical tasks (9).

The ability to generalize on distributions different from the training distribution—known as out-of-distribution (OOD) generalization—is well documented empirically, ranging from mathematical tasks that involve arithmetic or algebraic structures (10–12), to language reasoning problems requiring multistep inference (13, 14). Yet, less is known about when a model achieves OOD generalization and how it solves a compositional task with OOD data.

Our empirical investigations are motivated by the pioneering work on the induction head (IH) (15, 16), which is a component within the Transformer architecture. Our main contributions are as follows.

First, on the synthetic task of copying sequences of arbitrary patterns, a two-layer Transformer (TF) exhibits an abrupt emergence of subspace matching that accompanies OOD generalization between two Transformer layers, a phenomenon that echoes emergent abilities (9).

Second, on language reasoning tasks where LLMs infer the meanings of planted symbols, including examples of in-context learning, OOD generalization requires

Significance

Large language models (LLMs) appear to solve certain novel tasks with appropriately formatted prompts, which is an ability known as out-of-distribution generalization. However, the underlying mechanisms remain poorly understood. We conduct an extensive analysis of the internal structure of these models in both synthetic settings and common language tasks. Specifically, we examine how models complete patterns based on hidden rules, such as copying sequence segments and mapping words to symbols. We find that generalization abilities of these models depend crucially on compositional structures, which are internally achieved by aligning the principal subspaces of two network layers. The analysis provides insights into how LLMs learn certain rules and generalize in novel tasks.

Author affiliations: ^aState Key Laboratory of General Artificial Intelligence, Beijing Institute for General Artificial Intelligence, Beijing 100080, China; and ^bDepartment of Statistics, University of Wisconsin, Madison, WI 53706

Author contributions: J.S. and Y.Z. designed research; J.S., Z.X., and Y.Z. performed research; J.S., Z.X., and Y.Z. analyzed data; and Y.Z. wrote the paper.

The authors declare no competing interest.

This article is a PNAS Direct Submission.

Copyright © 2025 the Author(s). Published by PNAS. This article is distributed under [Creative Commons Attribution-NonCommercial-NoDerivatives License 4.0 \(CC BY-NC-ND\)](#).

¹To whom correspondence should be addressed. Email: yiqiao.zhong@wisc.edu.

This article contains supporting information online at <https://www.pnas.org/lookup/suppl/doi:10.1073/pnas.2417182122/-DCSupplemental>.

Published February 7, 2025.

a similar compositional structure. Extensive experiments on LLMs suggest the presence of a latent subspace for compositions in multilayer and multihead models, which we propose as the common bridge representation hypothesis.

An Exemplar: Copying. Copying is a simple yet nontrivial task that exemplifies OOD generalization. Briefly speaking, given a sequence that contains several consecutive tokens such as $[A]$, $[B]$, $[C]$, a model predicts the next token as $[C]$ upon receiving $[A]$, $[B]$:

$$\begin{array}{ccc} \dots [A], [B], [C] \dots [A], [B] & \xrightarrow{\text{next-token prediction}} & \\ \dots [A], [B], [C] \dots [A], [B], [C] & & \end{array}$$

Formally, consider a sequence of tokens $\mathbf{s} = (s_1, \dots, s_T)$ where each s_t is in a discrete vocabulary set $\mathcal{A}$. Suppose that a segment of length L is repeated in this sequence: $\mathbf{s}_{T_0:(T_0+L-1)} = \mathbf{s}_{(T-L+1):T}$, where $L < (T - T_0)/2 + 1$. Upon receiving the sequence $\mathbf{s}_{1:(T-1)}$, copying requires a model to output* token s_T .

Copying represents a primitive form of abstract reasoning. While humans can code the copying rule into a model, classical statistical models have difficulty learning this rule purely based on instances of such sequences. For instance, hidden Markov models and n -gram models (17) require estimating a large transition matrix or high-order conditional probability, which scales exponentially in L .

As a simple experiment, we fix $|\mathcal{A}| = 64$ and consider a power law distribution $\mathcal{P}$ on $\mathcal{A}$. We sample each sequence $\mathbf{s}$ of length $T_{\max} = 64$ independently by planting repeated segments in a “noisy background”:

1. Sample L uniformly from $\{10, 11, \dots, 19\}$, sample $s_t^\#$ from $\mathcal{P}$ independently for $t = 1, \dots, L$, and form a segment $\mathbf{s}^\# = (s_1^\#, \dots, s_L^\#)$;
2. Denote $r_L = T_{\max} - 2L$. Sample two integers uniformly from $\{1, 2, \dots, r_L\}$ and denote the smaller/larger ones by T_0, T_1 ;
3. Form a sequence $(*, \mathbf{s}^\#, *, \mathbf{s}^\#, *)$ where $*$ is filled by random tokens of lengths $T_0, T_1 - T_0, r_L - T_1$ respectively drawn from $\mathcal{P}$ independently.

We train a two-layer attention-only Transformer on batches of fresh samples, namely each training step uses independently drawn sequences. Model architecture and training follow the standard practice; see *SI Appendix, section 2* for details. We report both ID test errors and OOD test errors by computing the average token-wise prediction accuracy based on the second segment. Here, the OOD error is evaluated on sequences of a similar format but $\mathcal{P}$ is replaced by the uniform distribution $\mathcal{P}_{\text{ood}}$, and L is replaced by $L_{\text{ood}} = 25$. As a comparison, we train a one-layer attention-only Transformer using the same data.

Compositional Structure Is Integral to OOD Generalization. In Fig. 1, we observe that the TF experiences two phases. i) In the weak learning phase, the model learns the marginal token distribution $\mathcal{P}$ and predicts the most probable token irrespective of the structure of the entire sequence $\mathbf{s}$. This results in a decrease of the ID error but not the OOD error. ii) In the rule-learning phase, the model learns the copying rule and generalizes

reasonably well on both ID and OOD data. In contrast, one-layer Transformer only achieves weak learning. Note that the OOD error is smaller after training because the longer repetition segment means a larger signal strength.

Learning the copying rule requires the model to generalize on OOD instances in two aspects.

1. Generalize to a larger repetition length L_{ood} (aka length generalization).
2. Generalize to a different token distribution $\mathcal{P}_{\text{ood}}$.

Our analysis of training dynamics later suggests that the two layers of the Transformer play complementary roles: One layer specializes in processing positional information and another in token information. Composing the two layers yields OOD generalization.

1. Background

We briefly introduce Transformers, model interpretability, and OOD generalization. For illustrative purposes, we focus on core model modules and selectively present key concepts, deferring further discussion of related work to *An Exemplar: Copying* and implementation details to *SI Appendix*.

1.1. Transformers. Transformers are standard neural network architectures for processing text and other sequential data. While many variants of Transformers have been developed, their core components align with the original Transformer model proposed by Vaswani et al. (18). The first step of processing is tokenization, where a string $\mathbf{s}$ is converted into tokens. Each token is an element of a vocabulary $\mathcal{A}$, typically representing words, subwords, special symbols, and similar units. Then, these tokens are mapped to numerical vectors known as token embeddings. In the original formulation (18), positional information is encoded by a set of positional embedding vectors, and the sum of token and positional embeddings produce input vectors $\mathbf{x}_1, \dots, \mathbf{x}_T \in \mathbb{R}^d$ to a Transformer.

1.1.1. Transformer architecture. The input vectors are processed by multiple Transformer layers, which yield hidden states $\mathbf{x}_1^{(\ell)}, \dots, \mathbf{x}_T^{(\ell)} \in \mathbb{R}^d$ after each layer ℓ . Let $\mathbf{X} = [\mathbf{x}_1^{(\ell)}, \dots, \mathbf{x}_T^{(\ell)}]^\top \in \mathbb{R}^{T \times d}$ be the input vectors or the hidden states at a layer ℓ ($\ell = 0$ means the input vectors). The computations at each layer follow the generic formulas: for each $\ell = 1, 2, \dots, L$,

$$\begin{aligned} \mathbf{X} &\leftarrow \mathbf{X} + \text{MSA}(\mathbf{X}; \mathbf{W}^{(\ell)}) \\ \mathbf{X} &\leftarrow \mathbf{X} + \text{FFN}(\mathbf{X}; \tilde{\mathbf{W}}^{(\ell)}), \end{aligned} \quad [1]$$

where $\mathbf{W}^{(\ell)}$ and $\tilde{\mathbf{W}}^{(\ell)}$ contain all trainable parameters. Here, a crucial component is the multihead self-attention (MSA), which is a mapping $\text{MSA}(\mathbf{X}; \mathbf{W}) : \mathbb{R}^{T \times d} \rightarrow \mathbb{R}^{T \times d}$ given by

$$\text{MSA}(\mathbf{X}; \mathbf{W}) = \sum_{j=1}^H \overbrace{\text{Softmax}(\underbrace{\mathbf{X} \mathbf{W}_{\text{QK},j} \mathbf{X}^\top}_{\text{QK circuit reads and matches info from stream}})}^{\text{attention matrix}} \underbrace{\mathbf{X} \mathbf{W}_{\text{OV},j}^\top}_{\text{OV circuit writes and adds info to stream}}, \quad [2]$$

where $\mathbf{W}_{\text{QK},j}, \mathbf{W}_{\text{OV},j} \in \mathbb{R}^{d \times d}$, and $\mathbf{W}$ denotes the collection of all such matrices. The softmax operation applies to each row of $\mathbf{X} \mathbf{W}_{\text{QK},j} \mathbf{X}^\top$ and yields attention weights that are positive and

*Since Transformers output probabilities at test time, a predicted token $\hat{s}$ is the one that maximizes the conditional probability mass function $p_T(s|\mathbf{s}_{1:(T-1)})$.

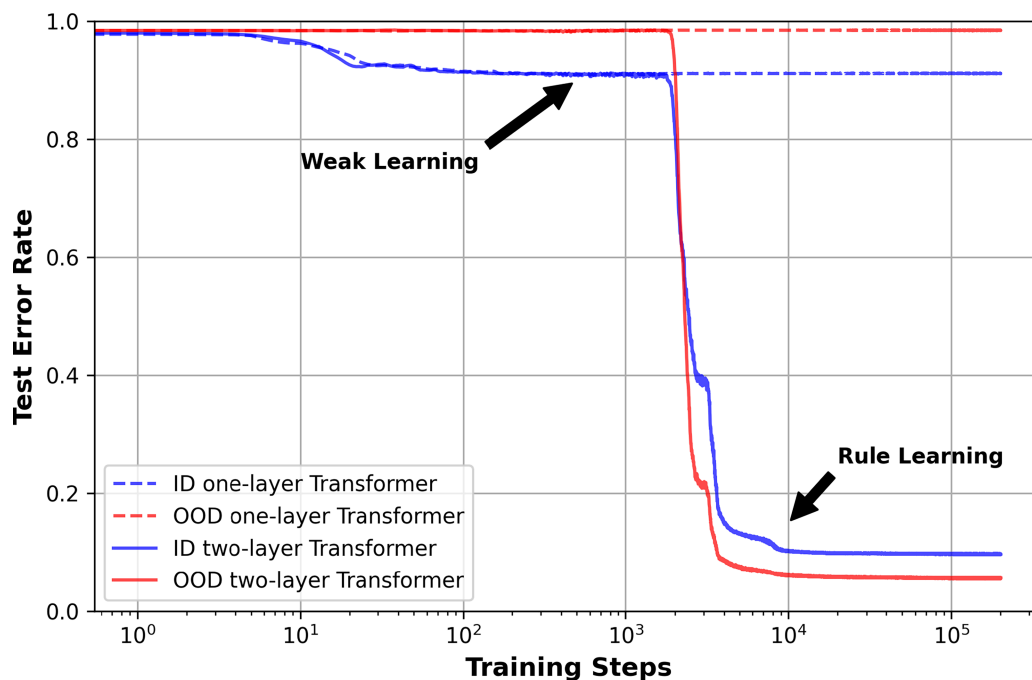


Fig. 1. Training a two-layer TF and a one-layer TF for copying task using **fresh samples** of the format $(*, s^{\#}, *, s^{\#}, *)$. The models are evaluated on an in-distribution (ID) test dataset and an OOD test dataset. **Weak learning phase:** The models rely on simple statistics of ID data and fail to generalize on OOD data; **Rule-learning phase:** two-layer TF learns the rule of copying from ID data and generalize well on ID/OOD data.

sum to 1 by definition. The feedforward network FFN applies to each column of $\mathbf{X}$ individually, and one common form reads $\text{FFN}(\mathbf{x}_t; \tilde{\mathbf{W}}) = \tilde{\mathbf{W}}_1 \text{ReLU}(\tilde{\mathbf{W}}_2 \mathbf{x}_t)$ for each position t .

At the final layer L , there is a simple classification layer $\text{Softmax}(\mathbf{W}^{(\text{out})} \mathbf{h}_T^{(L)})$, where $\mathbf{W}^{(\text{out})} \in \mathbb{R}^{|\mathcal{A}| \times d}$, to convert the last-position hidden states $\mathbf{h}_T^{(L)}$ to a probability vector, which is used to predict the next token. During pretraining, the entire model (weights in all layers and token embeddings) is trained with backpropagation.

1.1.2. Historical development. Word embedding (19–21), the precursor to Transformers, aims to represent words as numerical vectors, similar to the input vectors described earlier. To find contextualized representation of text strings, recurrent neural networks (RNNs) and long short-term memory networks (LSTMs) were initially employed, but it was soon found that attention mechanism (22, 23) handles long-range dependencies more effectively, laying the foundation for Transformers. As the scale of model sizes and pretraining corpora increased, Transformers started to exhibit surprising capabilities. Notably, they can generalize on many downstream tasks with crafted prompts without even updating the model parameters (7). For example, in in-context learning, a string of instructions or examples are added as a prefix to the test string, and the model seemingly “learns” from the context and generalizes on new tasks.

1.2. Interpreting Transformers. How do Transformers process and represent text data? While Transformers are designed and often used as black-box models, recent analyses offer partial insights into this question. The hidden states $\mathbf{x}_t$ are believed to encode the semantic meaning of input texts as a linear combination of base concept vectors, e.g., ref. 24

$$\begin{aligned} \text{apple} = & 0.09 \text{“dessert”} + 0.11 \text{“organism”} + 0.16 \text{“fruit”} \\ & + 0.22 \text{“mobile\&IT”} + 0.42 \text{“other”}, \end{aligned} \quad [3]$$

which means that the hidden states representing the word “apple” is decomposed into a linear combination of vectors of interpretable base concepts/features. A further refined decomposition of “other” vector may reveal syntactic features, positional features, and other contextual features.

This approach of interpreting Transformers is strongly supported by empirical evidence. In word embedding, word vectors show interpretable factor structure, such as “King – Man + Woman = Queen” (19); the FFN component is found to memorize concepts (25); and a dictionary learning algorithm applied to hidden states identifies a substantial number of features on large-scale language models (26, 27). These insights have inspired practice in model editing (28) and task arithmetic (29).

Below we present a list of key concepts often used for interpreting Transformers.

- **Linear representation hypothesis (LRH) and feature superposition.** The LRH posits that concepts are encoded as linear subspaces within the embedding or hidden states space. Feature superposition posits that hidden states are sparse linear combinations of base concept vectors from a potentially large dictionary.
- **Attention matrix.** It is the output of Softmax operation in each of H attention heads in Eq. 2. An attention matrix $\mathbf{A} \in \mathbb{R}^{T \times T}$ indicates how an attention head aggregates information from past tokens via the weighted sum, where the weights correspond to the entries of $\mathbf{A}$.
- **Residual stream.** It is the output of the identity map (namely a copy of $\mathbf{X}$) in Eq. 1, which is updated by the outputs of both MSA and FFN in each Transformer layer.
- **QK circuit.** It is a component of an attention head that computes $\mathbf{Z} = \mathbf{X} \mathbf{W}_{\text{QK},j} \mathbf{X}^T \in \mathbb{R}^{T \times T}$. A QK circuit provides a similarity measure between two hidden states, and $\mathbf{Z}$ decides at each token how much an attention head “reads” and “processes” information from past tokens.

- OV circuit. It is a component of an attention head that computes $\mathbf{A}\mathbf{X}\mathbf{W}_{\text{OV},j}^\top$, whose output is added to the residual stream. The OV circuit decides how information is “written” to the stream.
- Previous-token head (PTH). It refers to a type of attention head in a Transformer that attends to the previous token, namely $A_{t,t-1} \approx 1$, for a typical input text.
- IH. It refers to a type of attention head that attends to the token next to an identical token on an input string with repetition patterns.

In the literature, PTHs and IHs are usually qualitative characterization of attention patterns as they are input-dependent. In this paper, we identify PTHs and IHs from all attention heads by calculating and ranking scores based on desired attention patterns, cf. Eqs. 5 and 6.

1.3. OOD Generalization. Since GPT-3 and GPT-4, many research communities have shown significant interest in understanding why trained models generalize well on novel tasks. Broadly speaking, OOD generalization refers to the generalization behavior of models when training data have a different distribution compared with the test data. Unlike classical statistics topics such as extrapolation, distribution shift, and domain adaptation, pretrained LLMs can solve compositional tasks and seemingly learn from novel context. While mathematical characterization of OOD data for natural languages remains elusive, recent experimental investigations have offered insights. We highlight several representative approaches for generating OOD data.

1. Changing the length or the distribution of task strings. This is particularly useful for experiments with synthetic data, where the distribution of task strings can be easily modified (30–32).
2. Replacing parts of a text string with counterfactual statements or unnatural symbols. It is used for probing LLMs to test reliability or reveal the internal mechanism (33–36).
3. Constructing compositional tasks based on deductive rules. It is used for testing models on complex reasoning tasks to examine whether models have learned the latent rules (14).

However, prior studies either fail to probe the inner workings of Transformer models or lack systematic measurements across many models. Compared with existing papers, we provide in-depth measurements to probe model behavior during training dynamics, and comprehensive experiments across many pre-trained models, ranging from small Transformer such as GPT-2 (124M parameters) to larger LLMs such as Llama-3 (70B parameters). Moreover, our findings about subspace matching provide a clear mathematical characterization, especially in the context of compositionality of LLMs.

2. Dissecting sharp transition in synthetic example

As in *An Exemplar: Copying*, we train a minimally working Transformer for the copying task as a clean synthetic example, though we find similar results on larger models. In particular, the model has two self-attention layers with a single head and no MLPs. The architecture and training are standard, including residual connection, LayerNorm, RoPE, dropout, autoregressive training with the AdamW optimizer, and weight decay. In *Appendix*, we provide a list of notations, experimental details, and comparison with model variants.

2.1. Progress Measures. Under the circuits’ perspective in Eq. 2, each of the H attention head consists of a “reading” QK circuit, a “writing” OV circuit, and the softmax operation. Since $H = 1$ in our experiment, our goal is to study how the 1st-layer attention head interacts with the 2nd-layer head. To this end, we define several measurements.

1. Diagonal score. For 1st-layer $\mathbf{W}_{\text{OV}}$ and 2nd-layer $\mathbf{W}_{\text{QK}}$, we calculate $\mathbf{W}^{\text{QKOV}} = \mathbf{W}_{\text{QK}}\mathbf{W}_{\text{OV}} \in \mathbb{R}^{d \times d}$ and define $z = z(\mathbf{W}_{\text{QK}}, \mathbf{W}_{\text{OV}})$ as

$$z = \frac{\text{Ave}\left((\mathbf{W}_{ii}^{\text{QKOV}})_{i \leq d}\right) - \text{Ave}\left((\mathbf{W}_{ij}^{\text{QKOV}})_{i,j \leq d}\right)}{\text{Std}\left((\mathbf{W}_{ij}^{\text{QKOV}})_{i,j \leq d}\right)},$$

where Ave, Std mean taking average and SD, respectively.

This score measures the strength of diagonal entries. Roughly speaking, z is interpreted as the signal-to-noise ratio under the “ $\lambda \mathbf{I}_d + \text{noise}$ ” assumption on $\mathbf{W}^{\text{QKOV}}$.

2. Subspace matching score. Fix rank parameter $r = 10$. We calculate the top- r right singular vectors $\mathbf{U} \in \mathbb{R}^{d \times r}$ of 2nd-layer $\mathbf{W}_{\text{QK}}$ and top- r left singular vectors $\mathbf{V} \in \mathbb{R}^{d \times r}$ of 1st-layer $\mathbf{W}_{\text{OV}}$. The column linear span $\mathcal{P}_{\text{QK}} := \text{span}(\mathbf{U})$ (or similarly for $\mathcal{V}$) represents the principal subspace for reading (or writing) information. We calculate a generalized cosine similarity between the two subspaces.

$$\text{sim}(\mathcal{P}_{\text{QK}}, \mathcal{P}_{\text{OV}}) := \sigma_{\max}(\mathbf{U}^\top \mathbf{V}), \quad [4]$$

where $\sigma_{\max}$ denotes the largest singular value.

This score is equivalent to the regular cosine similarity between two optimally chosen unit vectors within the two subspaces. Results are analogous under a similar average similarity between $\mathcal{P}_{\text{QK}}, \mathcal{P}_{\text{OV}}$.

In Fig. 2 (top row), we identified simultaneous sharp transitions in generalization and also in the two scores. To investigate why the structural matching yields OOD generalization, we consider more measurements. An attention matrix $\mathbf{A} = (A_{t,t'})_{t,t' \leq T}$ is the output of the softmax in Eq. 2. The weight $A_{t,t'} \in [0, 1]$ represents the coefficient (aka attention) from position t to t' in a sequence and satisfies $\sum_{t'} A_{t,t'} = 1$.

3. PTH (previous-token head) and IH (induction-head) attention scores. Given an attention head and N input sequences, we first calculate the attention matrices $(\mathbf{A}_i)_{i \leq N}$. The PTH attention score measures the average attention to the previous adjacent token, and IH attention score measures the average attention to the to-be-copied token:

$$\text{score}^{\text{PTH}} = \text{Ave}_{i \leq N} \left(\frac{1}{T-1} \sum_{T \geq t \geq 2} (\mathbf{A}_i)_{t,t-1} \right), \quad [5]$$

$$\text{score}^{\text{IH}} = \text{Ave}_{i \leq N} \left(\frac{1}{|\mathcal{I}_i|} \sum_{t \in \mathcal{I}_i} (\mathbf{A}_i)_{t,t-L_i+1} \right), \quad [6]$$

where $\mathcal{I}_i$ is the index set of repeating tokens (second segment $\mathbf{s}^\#$), and L_i is the relative distance between two segments.

To emphasize the OOD performance, we calculate IH scores by using the OOD dataset of format $(*, \mathbf{s}^\#, *, \mathbf{s}^\#, *)$ described in *An Exemplar: Copying*. Moreover, PTH scores are based on random tokens uniformly drawn from the vocabulary as total removal of token-specific information.

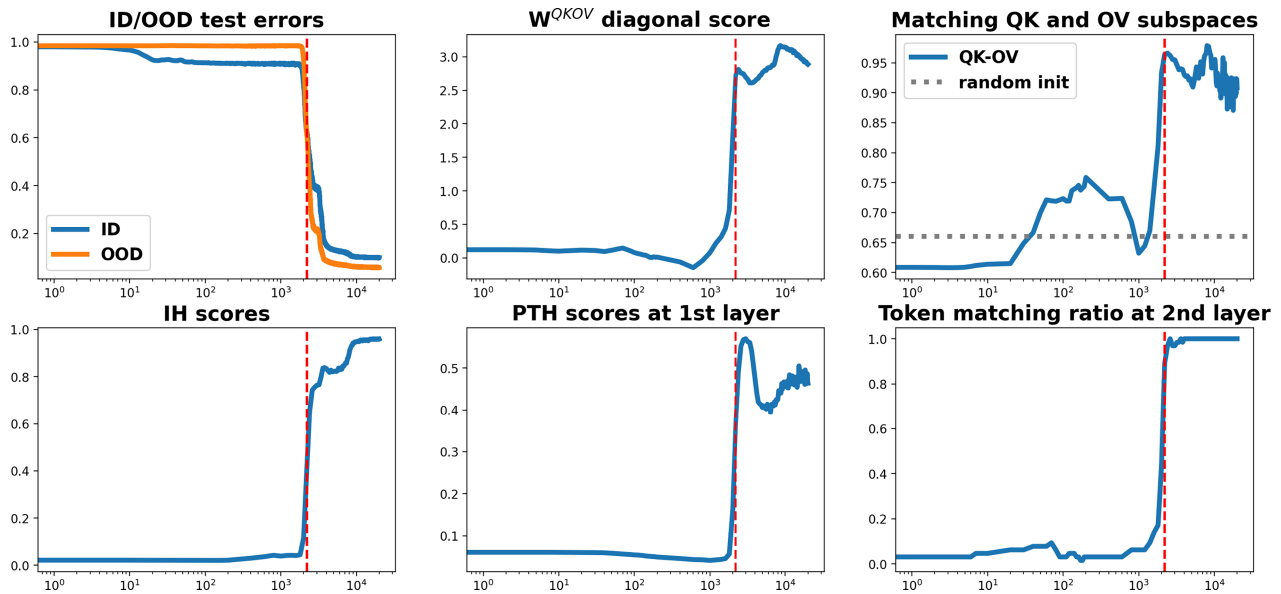


Fig. 2. Measuring training dynamics for two-layer one-head Transformers on the copying synthetic data. **First row:** Test errors drop abruptly as structural matching occurs. **Middle and Right** plots measure the matching between 1st-layer output circuit (OV) and 2nd-layer input circuit (QK). **Second row:** Model achieves OOD generalization by learning to compose two functionally distinct components (position matching vs. token matching). **Left** plot shows the formation of the IH on OOD data. **Middle** shows PTH scores on completely random tokens devoid of token info. **Right** shows token matching stripped of positional info.

- Token matching ratio. Let $\mathbf{e}_j \in \mathbb{R}^d$ be the token embedding for the j -th token in the vocabulary.

$$\text{Matching ratio} = \text{Ave}_{j \in \mathcal{A}} \left[\arg \max_{j' \leq j} (\mathbf{e}_j^\top \mathbf{W}^{QKOV} \mathbf{e}_{j'}) = j \right].$$

This ratio isolates pure token components $\mathbf{e}_j, \mathbf{e}_{j'}$ from embeddings and examines whether functionally, an identical token maximally activates the 2nd-layer attention through the path of OV–QK circuits.

Additionally, we consider the same copying task with slightly generalized data generation: we restrict the repeating patterns $\mathbf{s}^\#$ to a subset $\mathcal{S}$. To be more specific, given length L and size S , first the subset $\mathcal{S} \subset \mathcal{A}^L$ is determined by sampling the pattern S times independently according to Step 1 in *An Exemplar: Copying*; then each $\mathbf{s}^\#$ is drawn uniformly from $\mathcal{S}$. We call $\mathcal{S}$ the repetition pool, and $S = |\mathcal{S}|$ the pool size.

2.2. Results.

2.2.1. OOD generalization is accompanied by abrupt emergence of subspace matching. The top row of Fig. 2 shows that the sharp transition of generalization occurs at training steps around 2,000. In the meantime, the weight matrices in the two layers of the model exhibit a sudden increase of alignment: a large diagonal component in $\mathbf{W}^{QKOV}$ appears, and the two principal subspaces of $\mathbf{W}_{QK}, \mathbf{W}_{OV}$ change from being relatively random (similar to random initialization) to highly synchronized.

The structure of $\mathbf{W}^{QKOV}$ helps to match similar embeddings. Indeed, if we believe that $\mathbf{W}_{QK}\mathbf{W}_{OV}$ has the “ $\lambda \mathbf{I}_d$ + noise” structure, then the embedding $\mathbf{x}_t$ at position t satisfies $\mathbf{x}_t^\top \mathbf{W}_{QK}\mathbf{W}_{OV}\mathbf{y} \approx \mathbf{x}_t^\top \mathbf{y}$. To maximize this inner product, $\mathbf{y}$ of fixed length must be approximately aligned with $\mathbf{x}_t$, so embeddings similar to $\mathbf{x}_t$ tend to receive large attentions in the 2nd layer. Further, subspace matching between QK and OV shows that aligning the embeddings depends on the low-dimensional principal subspaces.

2.2.2. Two layers have complementary specialties: position shifting and token matching. The bottom row of Fig. 2 provides an

explanation for OOD generalization. The 1st-layer attention head has a large PTH score after training, even for sequences of completely random tokens. The high PTH score indicates that the 1st-layer head specializes in position shifting. In fact, in the ideal case where $A_{t,t-1} = 1$, the map $\mathbf{X} \mapsto \mathbf{A}\mathbf{X}$ is simply the shifting operator. Complementarily, the 2nd-layer QK matches the OV circuit and serves as token matching. So upon accepting the shifted tokens as inputs, the 2nd-layer head attends to the next position after the repeated token. Collectively, they yield an IH head, attending correctly to the to-be-copied token; see Fig. 3 (*Right*).

2.2.3. Memorization vs. generalization: Pattern diversity matters.

Fig. 3 shows that a smaller pool size S (decreasing from 1,000 to 750) requires more training steps to achieve good generalization. Moreover, when S drops below 740, models fail to generalize well under 20K steps; instead, they may memorize the repetition patterns, yielding small ID errors especially for small S . We also find that memorizing models exhibit very different attention matrices (*SI Appendix, Figs. S8 and S9*), suggesting the failure of learning the copying rule.

3. Intervention Experiments in LLMs

How is the synthetic example relevant to realistic reasoning tasks for LLMs? In this section, we address this question by presenting two types of realistic scenarios: OOD prompts (Sections 3.3 and 3.4) and realistic compositional structures (Section 3.5). Through these examples, we aim to demonstrate two key points:

- Prompts (natural language inputs) planted with arbitrarily chosen symbols can be inferred by LLMs in certain tasks without fine-tuning. This reasoning abilities depend crucially on IHs.
- Subspace matching as the compositional mechanism takes a more general form in multilayer and multihead models, where a shared latent subspace matches many PTHs and IHs simultaneously.

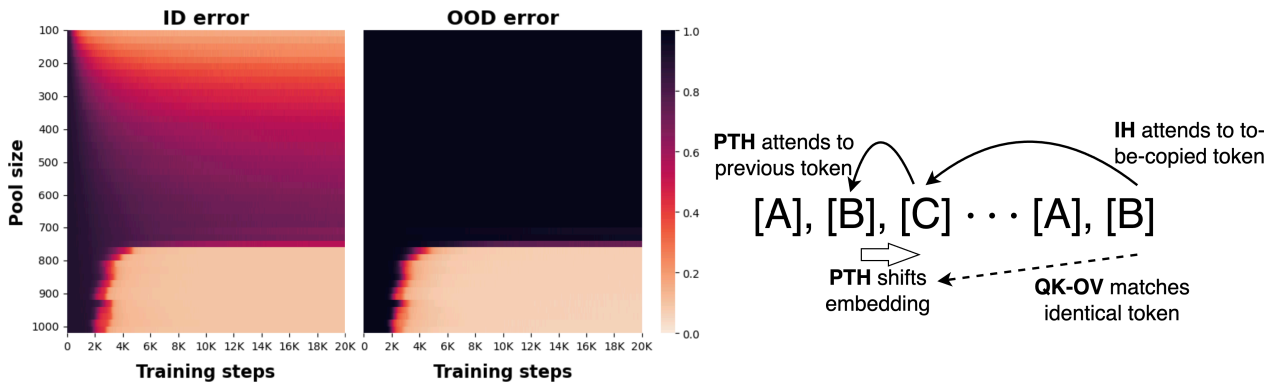


Fig. 3. Left: Memorization vs. generalization: More varied repetition patterns help models to learn the copying rule. When the set S of allowable $s^\#$ during training has a size smaller than 740, models fail to learn the rule and generalize OOD under 20K steps, yet they can still memorize the patterns if the pool size is small. Right: Composition of two layers expresses the rule of copying. 1st-layer head shifts the embedding at [B] to [C]. Through the QK-OV circuits, the embedding at [C] then matches the last token [B] in the 2nd-layer attention calculation, resulting in attention to [C] and completes the copying task.

3.1. Pretrained LLMs. We consider a variety of LLMs in our experiments: 1) Llama2-7B (37), 2) Llama3-8B (38), 3) Mistral-7B (39), 4) Falcon-7B (40), 5) Falcon2-11B (41), 6) OLMo-7B (42), 7) Gemma-7B (43), and 8) Gemma2-8B (44). See [SI Appendix](#) for details about models and implementations.

3.2. Defining IHs and previous-token heads. We sample $N = 100$ test prompts with a simple repetition pattern $(s^\#, s^\#)$: a block of 25 uniformly random tokens followed by a replica of the block, totaling $T = 50$ tokens. For any layer ℓ and head j of a Transformer, we denote by $A_i^{\ell,j} \in \mathbb{R}^{T \times T}$ the attention matrix defined in Eq. 2 on a test prompt i . By definition $\sum_{t'} (A_i^{\ell,j})_{t,t'}^{\ell,j} = 1$. This definition of IHs and PTHs only depends on the model, irrespective of downstream tasks.

For each model, we score all attention heads according to Eqs. 5 and 6 based on the test prompts, yielding $\text{score}_{\ell,j}^{\text{PTH}}$ and $\text{score}_{\ell,j}^{\text{IH}}$. Then we rank the PTH scores and IH scores in descending order respectively. For a prespecified K , we define PTHs (or IHs) as the top- K attention heads according to $\text{score}_{\ell,j}^{\text{PTH}}$ (or $\text{score}_{\ell,j}^{\text{IH}}$).

See [SI Appendix](#) for the lists of PTHs and IHs in each model and a set of complete examples we introduce below.

3.3. Symbolized Language Reasoning. In each task, we sample prompts based on a specified rule and use LLMs directly to predict next tokens. The last tokens are the target outputs for prediction.

1. Fuzzy copying: $[A], [B], [C] \dots [A'], [B'], [C']$

We consider conversion from lower-cased words to upper-cased words. For example, the correct completion of “bear snake fox poppy plate BEAR SNAKE FOX POPPY” is “PLATE.”

The IOI and ICL tasks below were proposed by Wang et al. (45) and Rong (33), and recently analyzed in refs. 35, 36, and 46–48. We extended the method in ref. 33 to construct symbolized prompts.

2. Indirect object identification (IOI):

[Subject] ... [Object] ... [Subject] ... [Object]

We sample input prompts where [Subject] and [Object] are common English names; see Fig. 4C for an example. Moreover, we consider a symbolized variant where the names of [Subject]

and [Object] are replaced by arbitrary special symbols. We select sets of symbols that are unconventionally combined so that the prompt instances are unlikely seen during training (thus OOD).

3. In-context learning (ICL): for f that maps an object to its category,

$$x_1, f(x_1), x_2, f(x_2) \dots x_n, f(x_n).$$

We consider mapping an object name x_i to one of the three categories {sport, plant, animal}. We concatenate instances that follow the format “object is category”; see Fig. 4C. Similarly, in the symbolized prompt, categories are replaced by special symbols. Again, the symbolized prompts are OOD because the three class labels are replaced by “unnatural” ones, requiring LLMs to infer their meanings at test time.

3.3.1. Experiments with removal of IHs. Given a prompt, we remove $K = 50$ IHs from a pretrained LLM by manually setting its attention matrix $A_i^{\ell,j}$ to a zero matrix. This effectively deletes the paths containing the targeted IHs from the architecture. As a baseline for comparison, we randomly select K pairs (ℓ, j) from all possible attention heads for deletion, repeated on five random seeds.

We measure the prediction accuracy for IOI and ICL in the multiple-choice form, namely we pick the solution with the highest prediction probability among candidate solutions: [Subject] and [Object] for IOI, and the three categories for ICL. We report prediction accuracy based on 100 sampled prompts for each task (Fig. 4). Accuracy by random guess is 1/2 and 1/3, respectively.

In addition, we examine how the accuracy changes across different scales of models (Fig. 5). We use the Pythia model suite (49), which is a collection of Transformers of varying sizes pretrained on the same data.

3.3.2. Results: Inferring from symbolized OOD prompts relies critically on IHs. Symbolized IOI/ICL instances represent a form of language reasoning, as they require combining semantic understanding with an extra step of converting names/labels to symbols. A key finding from Fig. 4D and E is that while LLMs generalize reasonably well on OOD prompts, removal of IHs significantly reduces prediction accuracy compared with removal of random heads.

Interestingly, LLMs on normal IOI and ICL prompts are very robust to IH removal. One explanation for the discrepancy between ID and OOD is that LLMs rely on memorized facts for

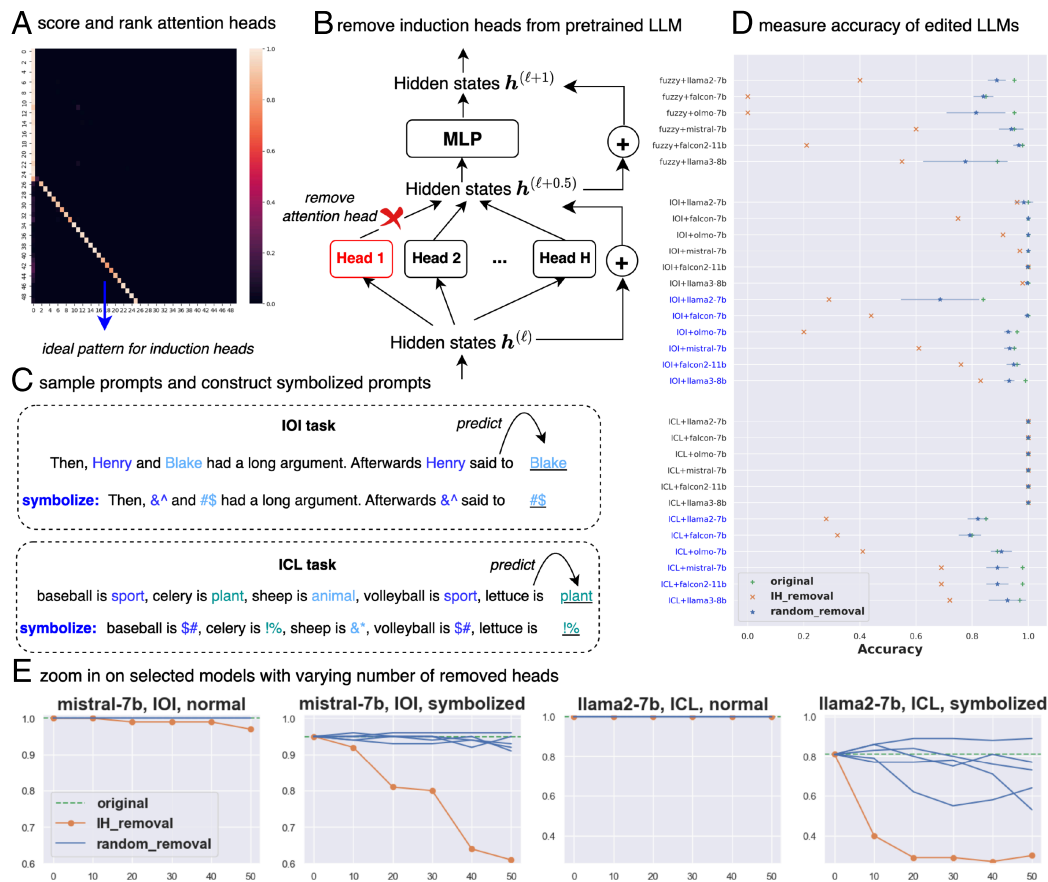


Fig. 4. LLMs depend on IHs for symbolized language reasoning. (A) We rank attention heads and determine IHs as $K = 50$ top-scoring heads. (B) We remove IHs by manually setting attention matrices to zero. (C) We sample instances according to the rule of each task and then construct OOD instances by symbolizing names/labels. (D) We measure the accuracy of various LLMs under IH removal. Symbolized tasks are indicated by blue names. We also report random baseline (deleting 50 randomly selected heads) using 5 random seeds, and report the variability using a segment showing $\pm$ one SD. (E) We show the accuracy vs. varying K , where a smaller K means deleting fewer heads.

ID prompts, and use combined abilities (both memorized facts and IHs) to solve OOD prompts.

3.4. Mathematical Reasoning with Chain-of-Thought. To further examine the impact of IHs in a more realistic setting,

we consider a popular benchmark dataset GSM8K (50), which contains a collection of K-12 mathematical questions. Solving the questions requires basic arithmetic abilities—usually a combination of addition, subtraction, multiplication, and division—on top of natural language understanding.

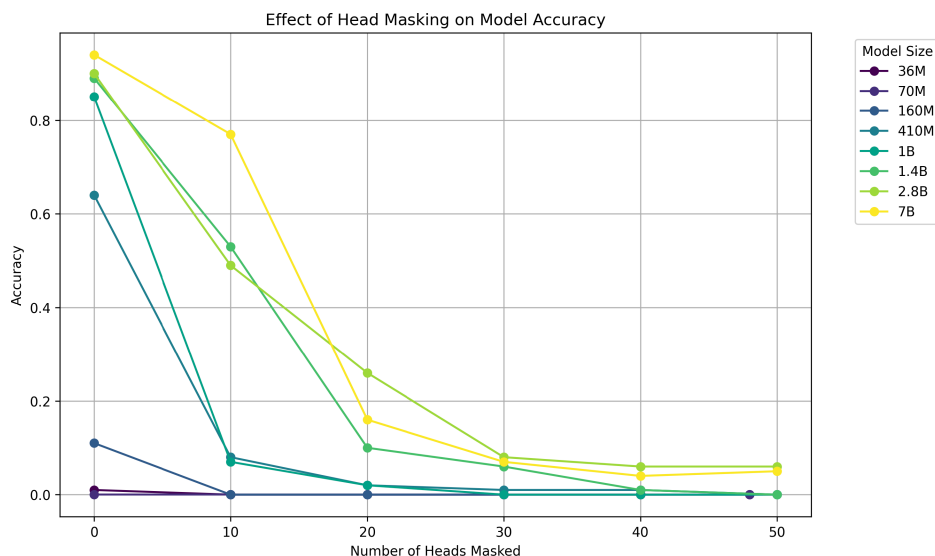


Fig. 5. Scaling experiment on fuzzy copying: how removal of IHs impact Pythia models. We use the family of Pythia models of varying sizes, ranging from 36M parameters to 7B parameters. As we increase the number of removed IHs, there is a consistent accuracy drop of all models on fuzzy copying.

GSM8K	GSM8K-rand Template
"Jerry is cutting up wood for his wood-burning stove. Each pine tree makes 80 logs, each maple tree makes 60 logs, and each walnut tree makes 100 logs. If Jerry cuts up 8 pine trees, 3 maple trees, and 4 walnut trees, how many logs does he get?"	"{name} is cutting up wood for his wood-burning stove. Each {pine} tree makes {pine_logs} logs, each {maple} tree makes {maple_logs} logs, and each {walnut} tree makes {walnut_logs} logs. If {name} cuts up {pine_count} {pine} trees, {maple_count} {maple} trees, and {walnut_count} {walnut} trees, how many logs does he get?"
"First find the total number of pine logs by multiplying the number of trees by the number of logs per tree: 80 logs/pine * 8 pines = <<80*8=640>>640 logs Then do the same thing for the maple trees: 60 logs/maple * 3 maples = <<60*3=180>>180 logs And do the same thing for the walnut trees: 100 logs/walnut * 4 walnuts = <<100*4=400>>400 logs Finally, add up the number of logs from each type of tree to find the total number: 640 logs + 180 logs + 400 logs = <<640+180+400=1220>>1220 logs #### 1220"	"First find the total number of {pine} logs by multiplying the number of trees by the number of logs per tree: {pine_logs} logs/{pine} * {pine_count} {pine} = <<{pine_logs}*{pine_count}={total_pine}>>{total_pine} logs Then do the same thing for the maple trees: {maple_logs} logs/{maple} * {maple_count} {maple} = <<{maple_logs}*{maple_count}={total_maple}>>{total_maple} logs And do the same thing for the walnut trees: {walnut_logs} logs/{walnut} * {walnut_count} {walnut} = <<{walnut_logs}*{walnut_count}={total_walnut}>>{total_walnut} logs Finally, add up the number of logs from each type of tree to find the total number: {total_pine} logs + {total_maple} logs + {total_walnut} logs = <<{total_pine}+{total_maple}+{total_walnut}={total}>>{total} logs #### {total}"

Fig. 6. An example from the GSM8K benchmark, and a template from GSM8K-rand that we constructed. *Left:* a sample question and its solution with CoT reasoning. The final solution is the number 1220 after ####. *Right:* We sample variables name, pine, maple, walnut independently from a pool of 40 symbol strings. Then we sample other variables such as pine_logs and pine_count independently from a range of integers. We ensure the reasoning logic leads to the correct solution total (This figure was created using tikz and does not have a separate source file).

Fig. 6 (*Left*) shows an example of a typical question and a solution with chain-of-thought (CoT) reasoning (51). CoT is a standard technique for LLMs to solve reasoning tasks, where intermediate steps are explicitly shown in in-context examples. Given the prompt that contains these in-context examples, LLMs are observed to be significantly better at solving reasoning tasks. We consider 10-shot learning, where 10 question-solution pairs are added as a prefix to a test question.

To generate explicit OOD test data, we constructed a variant of GSM8K which we call GSM8K-rand. This dataset comprises 12 templates derived from maintaining the core structures of 12 question-solution examples from GSM8K and randomizing names and numbers, as shown in Fig. 6 (*Right*). Inspired by ref. 52, we sample the numbers from a range of integers with 1 to 3 digits, and sample the names and items from a predefined pool. A key difference from ref. 52 is that the pool for names and items consists of 40 tuples of special symbols such as "\$#", "@%", "!&," "*,", "#@", so the test examples were unlikely seen by the LLMs during training. Similar to the previous three tasks, we measure the accuracy of models under the removal of IHs. *SI Appendix, section 4* details the template construction and measurements.

3.4.1. Results: CoT reasoning relies heavily on IHs. Similar to Fig. 4, we remove IHs and measure the accuracy of three LLMs on GSM8K and GSM8K-rand. Table 1 shows that IHs have a large impact on model accuracy on both datasets, as their removal significantly decreases the model accuracy compared with the removal of the same number of random heads. One possible explanation is that CoT reasoning depends on copying abilities to reuse intermediate outputs in its subsequent calculation. We leave a comprehensive investigation to future work.

3.5. Common Bridge Representation Hypothesis. How do compositions work in LLMs beyond the synthetic example? We consider four tasks: i) Copying, ii) Fuzzy Copying, iii) IOI, and iv) ICL. For illustrative purposes, we report results for the copying task. The results for the other three tasks are similar, which can be found in *SI Appendix, section 6*. With a microscopic analysis, we posit the following common bridge representation (CBR) hypothesis.

For compositional tasks, a latent subspace stores intermediate representations from the outputs of relevant attention heads and then matches later heads.

The CBR hypothesis is an extension of linear representation hypothesis to compositional tasks. We give a mathematical statement of the CBR hypothesis in the ideal scenario. For a matrix $\mathbf{W}_0 \in \mathbb{R}^{d \times d}$, we denote by $\text{span}(\mathbf{W}_0) := \{\mathbf{W}_0 \mathbf{x} : \mathbf{x} \in \mathbb{R}^d\}$ the column span of $\mathbf{W}_0$. Let $\mathbf{W}_{\text{OV},j}, \mathbf{W}_{\text{QK},k} \in \mathbb{R}^{d \times d}$ be the matrices in a task-relevant OV/QK circuit in an earlier/layer respectively. Then, for a compositional task, there exists a low-dimensional subspace $\mathcal{V} \subset \mathbb{R}^d$ such that

$$\mathcal{V} = \text{span}(\mathbf{W}_{\text{OV},j}) = \text{span}(\mathbf{W}_{\text{QK},k}^T). \quad [7]$$

Intuitively, the output of an OV circuit has the form $\mathbf{W}_{\text{OV},j} \mathbf{x}$, which lies in $\mathcal{V}$. It needs to be processed by a later QK circuit through $\mathbf{W}_{\text{QK},k}$ to match relevant hidden states.

LLMs in practice are trained with stochastic gradient descent, so we do not expect Eq. 7 to hold exactly. Nevertheless, it motivates us in our experiment to estimate $\mathcal{V}$ from a collection of relevant QK circuits: given $(\mathbf{W}_{\text{QK},k})_{k=1}^K$, we apply singular value

Table 1. CoT reasoning depends on IHs on mathematical tasks

Accuracy		Number of IHs/Random heads removed					
		0	10	20	30	40	50
Llama2-7B	GSM8K	0.11	0.06/0.12	0.01/0.11	0.01/0.09	0.02/0.07	0.01/0.07
	GSM8K-rand	0.29	0.23/0.27	0.10/0.25	0.08/0.19	0.06/0.20	0.08/0.16
Llama2-70B	GSM8K	0.59	0.41/0.55	0.21/0.52	0.21/0.52	0.09/0.53	0.01/0.50
	GSM8K-rand	0.87	0.68/0.81	0.53/0.80	0.51/0.80	0.29/0.78	0.10/0.76
Llama3-70B	GSM8K	0.89	0.88/0.90	0.92/0.90	0.86/0.89	0.60/0.88	0.51/0.87
	GSM8K-rand	0.96	0.96/0.96	0.95/0.97	0.61/0.98	0.40/0.97	0.51/0.94

Accuracy comparison under removal of IHs vs. random heads (averaged over five random seeds).

decomposition to the stacked matrix $[W_{QK,1}, \dots, W_{QK,K}]$ and use the principal right singular subspace as an estimate of $\mathcal{V}$.

We experiment on a variety of LLMs and highlight key findings in Fig. 7 B–F, where we use GPT-2 as a recurring example and summarize results of other models. Top-scoring PTHs and IHs are distributed across different layers of LLMs, though more PTHs appear in early layers. We sample sequences of the format $(s^\#, s^\#, s^\#)$ and calculate the average token-wise probability/accuracy for predicting the third segment $s^\#$. See experiment details in SI Appendix.

3.5.1. Experiments with two interventions. For both experiments, we apply screening to top PTHs and IHs based on diagonal

scores, resulting in an average of nine effective PTHs and seven effective IHs. The first intervention experiment involves shuffling heads. We randomly permute matrices W_{QK} within the list of IHs, yielding an edited model (“shuffle within”). As comparison, we replace each W_{QK} within the list by a random W_{QK} outside the list, yielding another edited model (“shuffle outside”). In a parallel experiment, we shuffle W_{OV} circuits within PTHs similarly. We evaluate the original model and edited models by calculating the average probability of predicting correct tokens.

The second intervention experiment involves projection of weight matrices based on the discussion on Eq. 7. First, we stack the QK matrices from IHs into a matrix $[W_{QK}^1, \dots, W_{QK}^K]$

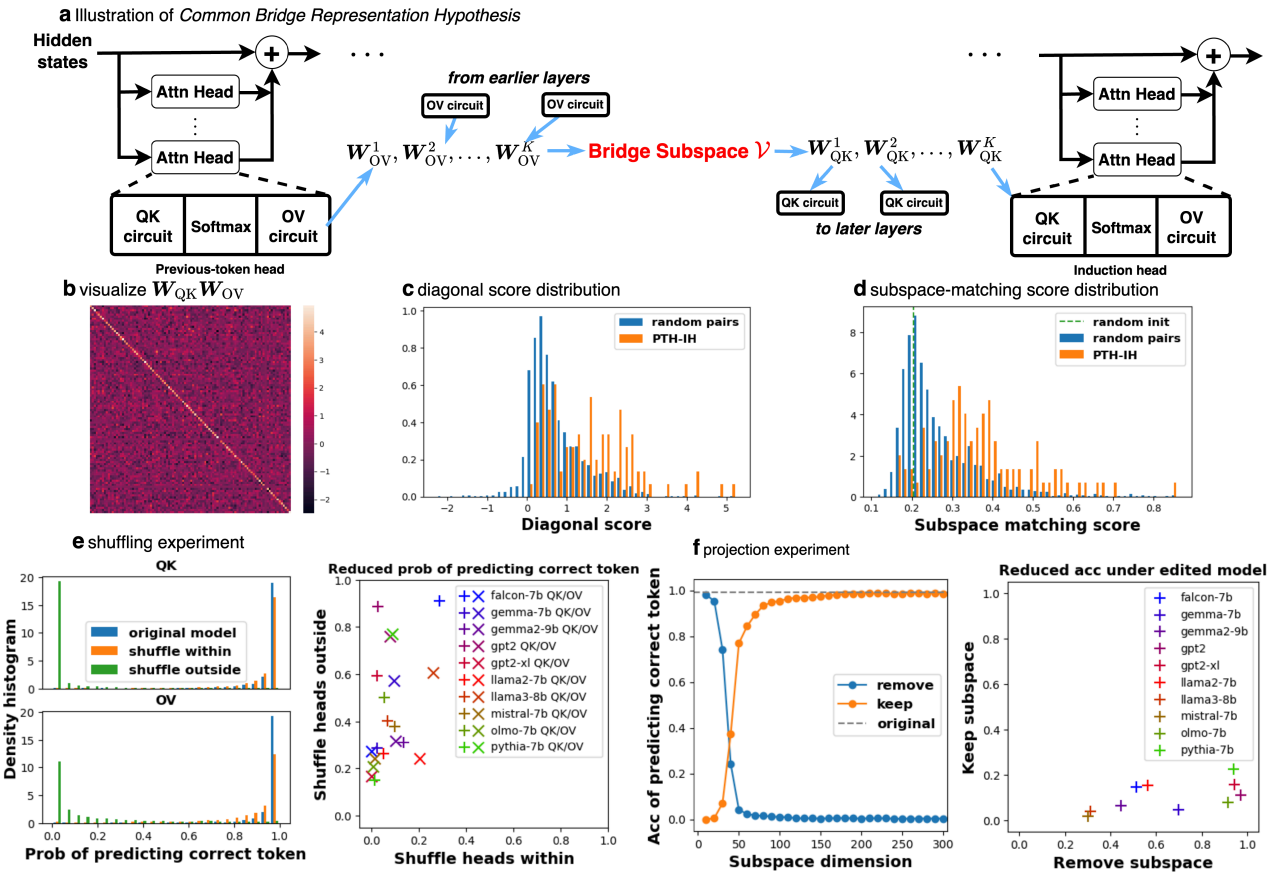


Fig. 7. Common bridge representation hypothesis as the mechanism of composition in LLMs. (A) Linear maps from circuits in relevant attention heads are connected by an intermediate *bridge subspace*. (B) We visualize $W_{QK} W_{OV}$ using highest-ranking PTH and IH. (C) Distributions of diagonal scores, comparing pairs from top-10 PTHs/IHs and pairs from random QK/OV circuits. (D) Distributions of subspace matching scores. (E) Shuffling W_{QK} (or W_{OV}) among top-scoring attention heads mildly changes probabilities of correct prediction, whereas replacing these QK or OV matrices with random heads significantly reduces the probabilities. (F) Removing a low-rank ($r = 50$ for GPT-2, $r = \frac{1}{20}d$ for others) bridge subspace from attention calculation (“remove”) heavily reduces prediction accuracy, whereas only using this subspace for attention calculation (“keep”) yields mild loss of accuracy.

and extract the top right singular vectors $V \in \mathbb{R}^{d \times r}$ for a prespecified r . We call the column linear span of V as the bridge subspace. Then, we edit 25% of all attention heads by weight projection $W_{QK} \leftarrow W_{QK} V V^\top$. After the edit (“keep”), the attention calculation can only use the component of embeddings within the bridge subspace. In a parallel experiment, we make a projection edit (“remove”) with an orthogonal complement $W_{QK} \leftarrow W_{QK} (I_d - V V^\top)$ to force attention calculation not to use component in the bridge subspace. We evaluate the edited models by calculating the average accuracy of predicting correct tokens.

3.5.2. Results: Subspace matching is a pervasive mechanism of compositions. Fig. 7B and C show that $W_{QK} W_{OV}$ contains large diagonal entries when the circuits are selected from PTHs and IHs. Further, Fig. 7D shows that leading (*Left*) singular spaces of W_{OV} and (*Right*) singular spaces of W_{QK} are highly aligned. Among many PTHs and IHs across different layers, subspace matching is a pervasive mechanism, which extends the findings from the synthetic example to realistic models.

3.5.3. Results: Relevant attention heads share a common latent subspace. We further show that the pairwise matching between circuits is not a coincidence; instead, a global latent subspace provides a “bridge” connecting relevant OV/QK circuits. The result of the shuffling experiment in Fig. 7E indicates that W_{QK} (or W_{OV}) from IHs (or PTHs) are approximately exchangeable, since permuting W_{QK} (or W_{OV}) does not reduce prediction probabilities significantly. Moreover, Fig. 7F further supports that V contains nearly complete information about copying, because removing/keeping this low-rank subspace from attention calculation drastically changes prediction accuracy in opposite directions.

The common bridge subspace is connected to the statistical literature on spiked matrices (53). Roughly speaking, the principal subspaces of W_{OV}^j , W_{QK}^j correspond to a shared spike, which contains relevant information for the compositional task.

4. Related Work

There are many recent papers on analyzing and interpreting Transformers (54–57); see ref. 58 for a comprehensive survey. We highlight a few key threads of research.

4.1. Mechanistic Interpretability and IHs. Mechanistic interpretability (MI) aims to provide microscopic interpretability of inner workings of Transformers (59). In particular, refs. 15 and 16 proposed IHs as crucial components of Transformers. In particular, they suggested that matching OV circuits and QK circuits is crucial for ICL. A line of empirical research extends IHs in various aspects (45, 60–66), and ref. 67 provides theoretical analysis of IHs assuming a latent causal graph. Compared with the existing literature, we conducted extensive experiments on both small Transformers and a variety of LLMs, demonstrating that IHs are crucial components for many reasoning tasks. Further, we propose that subspace matching—and more broadly common bridge representation hypothesis—as the compositional mechanism.

4.2. OOD Generalization and Compositions. Historically, studies of generalization on novel domains focus on extrapolation, distribution shift, domain adaptation, etc. Since GPT-3 (5), recent studies of OOD generalization focus on arithmetic and algebraic tasks (11, 30), formal language and deductive reasoning (68–71), learning Boolean functions (12), etc. Our copying task is an example of length generalization (72, 73). Compositional tasks are strongly related to reasoning abilities of LLMs, such as arithmetic tasks (74, 75), formal language (76), logical rules (77, 78), and so on. Despite recent insights (74, 79–81), there is a lack of systematic and quantitative analysis. Our work is a step toward understanding compositional capabilities of LLMs in a systematic way.

4.3. Linear Representation Hypothesis. Linear representation hypothesis (LRH) states that monosemantic (meaning basic or atomic) concepts are represented by linear subspaces (or half-spaces) in embeddings (82–84). This hypothesis is empirically verified not only in simpler statistical models but also in LLMs (20, 85). The LRH treats subspaces as fundamental units for representing linguistic concepts, but it does explain how models solve reasoning tasks or achieve OOD generalization. Our CBR hypothesis furthers this view by linking intermediate outputs in compositions to interpretable subspaces.

5. Limitations and Future Work

First, our hypothesis is a general conjecture on the mechanism of compositions in Transformers, based on our analysis of IHs. While IHs are pervasive in LLMs, other components or mechanisms for compositions may exist. Additionally, our interpretations are based on a simplified form of self-attention. It would be interesting to explore alternative mechanisms for compositions, and examine variants or practical techniques in LLMs that may impact our hypothesis.

Second, we did not develop insights to explain the emergence of the bridge subspace during training. The sharp transition in prediction accuracy is related to the emergent abilities of LLMs observed in broader contexts (9). In simpler settings, feedforward networks learning algebraic rules also exhibit phase transitions from memorization to generalization, a phenomenon known as Grokking (86–91). Analyzing the training dynamics of Transformers for copying and other compositional tasks would be an interesting avenue for further research.

Data, Materials, and Software Availability. The code for replicating the experiments and analyses can be found in GitHub page: <https://github.com/JiajunSong629/ood-generalization-via-composition> (92). Previously published data were used for this work (HuggingFace).

ACKNOWLEDGMENTS. This work was partially supported by NSF Division of Mathematical Sciences Grant 2412052 and by the Office of the Vice Chancellor for Research and Graduate Education at the UW Madison with funding from the Wisconsin Alumni Research Foundation. We are grateful for the feedback from Yingyu Liang, Haolin Yang, Junjie Hu, and Robert Nowak.

1. S. Bubeck *et al.*, Sparks of artificial general intelligence: Early experiments with gpt-4. *arXiv [Preprint]* (2023). <https://arxiv.org/abs/2303.12712> (Accessed 22 May 2023).
2. P. Liang *et al.*, Holistic evaluation of language models. *arXiv [Preprint]* (2023). <https://doi.org/10.48550/arXiv.2211.09110> (Accessed 16 November 2022).
3. D. Donoho, Data science at the singularity. *Harv. Data Sci. Rev.* **6**, (2024), <https://hdsr.mitpress.mit.edu/pub/9g9mau4m0>.
4. Y. Bengio, N. Malkin, Machine learning and information theory concepts towards an AI mathematician. *Bull. Am. Math. Soc.* **61**, 457–469 (2024).

5. T. Brown *et al.*, Language models are few-shot learners. *Adv. Neural Inf. Process. Syst.* **33**, 1877–1901 (2020).
6. Q. Dong *et al.*, A survey for in-context learning. *arXiv [Preprint]* (2022). <https://arxiv.org/abs/2301.00234> (Accessed 31 December 2022).
7. L. Reynolds, K. McDonell, “Prompt programming for large language models: Beyond the few-shot paradigm” in *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems* (2021), pp. 1–7.
8. T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, Y. Iwasawa, Large language models are zero-shot reasoners. *Adv. Neural Inf. Process. Syst.* **35**, 22199–22213 (2022).

9. J. Wei *et al.*, Emergent abilities of large language models. *arXiv [Preprint]* (2022). <https://arxiv.org/abs/2206.07682> (Accessed 9 June 2022).
10. A. Kazemnejad, I. Padhi, K. Natesan, P. Das, S. Reddy, "The impact of positional encoding on length generalization in transformers" in *Thirty-Seventh Conference on Neural Information Processing Systems* (2023).
11. Y. Zhang *et al.*, Unveiling transformers with lego: A synthetic reasoning task. *arXiv [Preprint]* (2022). <https://arxiv.org/abs/2206.04301> (Accessed 9 June 2022).
12. E. Abbe, S. Bengio, A. Lotfi, K. Rizk, "Generalization on the unseen, logic reasoning and degree curriculum" in *International Conference on Machine Learning* (PMLR, 2023), pp. 31–60.
13. Z. Wu *et al.*, "Reasoning or reciting? Exploring the capabilities and limitations of language models through counterfactual tasks" in *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, K. Duh, H. Gomez, S. Bethard, Eds. (Association for Computational Linguistics, Mexico City, Mexico, 2024), pp. 1819–1862.
14. A. Saparov *et al.*, Testing the general deductive reasoning capacity of large language models using OOD examples. *Adv. Neural Inf. Process. Syst.* **36**, 3083–3105 (2024).
15. N. Elhage *et al.*, A mathematical framework for transformer circuits. *Transformer Circuits Thread* (2022). <https://transformer-circuits.pub/2021/framework/index.html>. Accessed 1 December 2021.
16. C. Olsson *et al.*, In-context learning and induction heads. *Transformer Circuits Thread* (2022). <https://transformer-circuits.pub/2022/in-context-learning-and-induction-heads/index.html>. Accessed 24 September 2022.
17. T. Brants, A. Popat, P. Xu, F. J. Och, J. Dean, "Large language models in machine translation" in *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)* (2007), pp. 858–867.
18. A. Vaswani *et al.*, Attention is all you need. *Adv. Neural Inf. Process. Syst.* **30**, 6000–6010 (2017).
19. T. Mikolov, Wt. Yih, G. Zweig, "Linguistic regularities in continuous space word representations" in *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (2013), pp. 746–751.
20. T. Mikolov, K. Chen, G. S. Corrado, J. Dean, "Efficient estimation of word representations in vector space" in *International Conference on Learning Representations* (2013).
21. J. Pennington, R. Socher, C. D. Manning, "Glove: Global vectors for word representation" in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (2014), pp. 1532–1543.
22. O. Vinyals, A. Toshev, S. Bengio, D. Erhan, "Show and tell: A neural image caption generator" in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2015), pp. 3156–3164.
23. D. Bahdanau, Neural machine translation by jointly learning to align and translate. *arXiv [Preprint]* (2014). <https://arxiv.org/abs/1409.0473> (Accessed 19 May 2016).
24. Z. Yun, Y. Chen, B. A. Olshausen, Y. LeCun, Transformer visualization via dictionary learning: contextualized embedding as a linear superposition of transformer factors. *arXiv [Preprint]* (2021). <https://arxiv.org/abs/2103.15949> (Accessed 29 March 2021).
25. M. Geva, R. Schuster, J. Berant, O. Levy, "Transformer feed-forward layers are key-value memories" in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing* (2021), pp. 5484–5495.
26. A. Templeton *et al.*, Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. *Transformer Circuits Thread* (2024). <https://transformer-circuits.pub/2024/scaling-monosemanticity/>.
27. L. Gao *et al.*, Scaling and evaluating sparse autoencoders. *arXiv [Preprint]* (2024). <https://arxiv.org/abs/2406.04093>. (Accessed 6 June 2024).
28. K. Li, O. Patel, F. Viégas, H. Pfister, M. Wattenberg, Inference-time intervention: Eliciting truthful answers from a language model. *Adv. Neural Inf. Process. Syst.* **36**, 41451–41530 (2024).
29. G. Ilharco *et al.*, Editing models with task arithmetic. *arXiv [Preprint]* (2022). <https://arxiv.org/abs/2212.04089> (Accessed 8 December 2022).
30. N. Lee, K. Sreenivasan, J. D. Lee, K. Lee, D. Papailiopoulos, Teaching arithmetic to small transformers. *arXiv [Preprint]* (2023). <https://arxiv.org/abs/2307.03381> (Accessed 7 July 2023).
31. A. Kazemnejad, I. Padhi, K. Natesan Ramamurthy, P. Das, S. Reddy, The impact of positional encoding on length generalization in transformers. *Adv. Neural Inf. Process. Syst.* **36**, 24892–24928 (2024).
32. B. Wang, X. Yue, Y. Su, H. Sun, Grokked transformers are implicit reasoners: A mechanistic journey to the edge of generalization. *arXiv [Preprint]* (2024). <https://arxiv.org/abs/2405.15071> (Accessed 23 May 2024).
33. F. Rong, "Extrapolating to unnatural language processing with gpt-3's in-context learning: The good, the bad, and the mysterious" *The Stanford AI Lab Blog* (2021). <http://ai.stanford.edu/blog/in-context-learning>. Accessed 28 May 2021.
34. Z. Wu *et al.*, Reasoning or reciting? Exploring the capabilities and limitations of language models through counterfactual tasks. *arXiv [Preprint]* (2023). <https://arxiv.org/abs/2307.02477> (Accessed 5 July 2023).
35. J. Pan, T. Gao, H. Chen, D. Chen, "What in-context learning 'learns' in-context: Disentangling task recognition and task learning" in *Findings of the Association for Computational Linguistics: ACL 2023*, A. Rogers, J. Boyd-Graber, N. Okazaki, Eds. (Association for Computational Linguistics, Toronto, ON, Canada) (2023), pp. 8298–8319.
36. D. Halawi, J. S. Denain, J. Steinhardt, "Overthinking the truth: Understanding how language models process false demonstrations" in *The Twelfth International Conference on Learning Representations* (2024).
37. H. Touvron *et al.*, Llama 2: Open foundation and fine-tuned chat models. *arXiv [Preprint]* (2023). <https://arxiv.org/abs/2307.09288> (Accessed 18 July 2023).
38. A. Dubey *et al.*, The llama 3 herd of models. *arXiv [Preprint]* (2024). <https://arxiv.org/abs/2407.21783> (Accessed 31 July 2024).
39. A. Q. Jiang *et al.*, Mistral 7b. *arXiv [Preprint]* (2023). <https://arxiv.org/abs/2310.06825> (Accessed 10 October 2023).
40. E. Almazrouei *et al.*, The falcon series of open language models. *arXiv [Preprint]* (2023). <https://arxiv.org/abs/2311.16867> (Accessed 28 November 2023).
41. Q. Malartic *et al.*, Falcon2-11b technical report. *arXiv [Preprint]* (2024). <https://arxiv.org/abs/2407.14885> (Accessed 20 July 2024).
42. D. Groeneveld *et al.*, OLMo: Accelerating the science of language models. *arXiv [Preprint]* (2024). <https://arxiv.org/abs/2402.00838> (Accessed 1 February 2024).
43. G. Team *et al.*, Gemma: Open models based on Gemini research and technology. *arXiv [Preprint]* (2024). <https://arxiv.org/abs/2403.08295> (Accessed 13 March 2024).
44. G. Team *et al.*, Gemma 2: Improving open language models at a practical size. *arXiv [Preprint]* (2024). <https://arxiv.org/abs/2408.00118> (Accessed 31 July 2024).
45. K. R. Wang, A. Variengien, A. Conmy, B. Shlegeris, J. Steinhardt, "Interpretability in the wild: A circuit for indirect object identification in GPT-2 small" in *The Eleventh International Conference on Learning Representations* (2023).
46. J. Wei *et al.*, Larger language models do in-context learning differently. *arXiv [Preprint]* (2024). <https://arxiv.org/abs/2303.03846> (Accessed 8 March 2023).
47. R. Agarwal *et al.*, "Many-shot in-context learning" in *ICML 2024 Workshop on In-Context Learning* (2024).
48. Z. Shi, J. Wei, Z. Xu, Y. Liang, "Why larger language models do in-context learning differently?" in *Forty-First International Conference on Machine Learning* (2024).
49. S. Biderman *et al.*, "Pythia: A suite for analyzing large language models across training and scaling" in *International Conference on Machine Learning* (PMLR, 2023), pp. 2397–2430.
50. K. Cobbe *et al.*, Training verifiers to solve math word problems. *arXiv [Preprint]* (2021). <https://arxiv.org/abs/2110.14168> (Accessed 27 October 2021).
51. J. Wei *et al.*, Chain-of-thought prompting elicits reasoning in large language models. *Adv. Neural Inf. Process. Syst.* **35**, 24824–24837 (2022).
52. I. Mirzadeh *et al.*, GSM-symbolic: Understanding the limitations of mathematical reasoning in large language models. *arXiv [Preprint]* (2024). <https://arxiv.org/abs/2410.05229> (Accessed 7 October 2024).
53. I. M. Johnstone, On the distribution of the largest eigenvalue in principal components analysis. *Ann. Stat.* **29**, 295–327 (2001).
54. R. Zhang, S. Frei, P. L. Bartlett, Trained transformers learn linear models in-context. *J. Mach. Learn. Res.* **25**, 1–55 (2024).
55. Y. Bai, F. Chen, H. Wang, C. Xiong, S. Mei, Transformers as statisticians: Provable in-context learning with in-context algorithm selection. *Adv. Neural Inf. Process. Syst.* **36**, 57125–57211 (2024).
56. J. Song, Y. Zhong, Uncovering hidden geometry in transformers via disentangling position and context. *arXiv [Preprint]* (2023). <https://arxiv.org/abs/2310.04861> (Accessed 7 October 2023).
57. A. R. Hsu, Y. Cherapanamjeri, A. Y. Odisho, P. R. Carroll, B. Yu, Mechanistic interpretation through contextual decomposition in transformers. *arXiv [Preprint]* (2024). <https://arxiv.org/abs/2407.00886> (Accessed 7 October 2023).
58. U. Anwar *et al.*, Foundational challenges in assuring alignment and safety of large language models. *arXiv [Preprint]* (2024). <https://arxiv.org/abs/2404.09932> (Accessed 15 April 2024).
59. C. Olah *et al.*, Zoom in: An introduction to circuits. *Distill* **5**, e00024-001 (2020).
60. G. Reddy, The mechanistic basis of data dependence and abrupt learning in an in-context classification task. *arXiv [Preprint]* (2023). <https://arxiv.org/abs/2312.03002> (Accessed 3 December 2023).
61. J. Merullo, C. Eickhoff, E. Pavlick, Circuit component reuse across tasks in transformer language models. *arXiv [Preprint]* (2023). <https://arxiv.org/abs/2310.08744> (Accessed 12 October 2023).
62. A. K. Singh, T. Moskovitz, F. Hill, S. C. Chan, A. M. Saxe, What needs to go right for an induction head? A mechanistic study of in-context learning circuits and their formation. *arXiv [Preprint]* (2024). <https://arxiv.org/abs/2404.07129> (Accessed 10 April 2024).
63. J. Ferrando, G. Sarti, A. Bisazza, M. R. Costa-jussà, A primer on the inner workings of transformer-based language models. *arXiv [Preprint]* (2024). <https://arxiv.org/abs/2405.00208> (Accessed 30 April 2024).
64. E. Akyürek, B. Wang, Y. Kim, J. Andreas, In-context language learning: Architectures and algorithms. *arXiv [Preprint]* (2024). <https://arxiv.org/abs/2401.12973> (Accessed 23 January 2024).
65. R. Gould, E. Ong, G. Ogden, A. Conmy, "Successor heads: Recurring, interpretable attention heads in the wild" in *The Twelfth International Conference on Learning Representations* (2024).
66. E. Akyürek, B. Wang, Y. Kim, J. Andreas, In-context language learning: Architectures and algorithms. *arXiv [Preprint]* (2024). <https://doi.org/10.48550/arXiv.2401.12973> (Accessed 23 January 2024).
67. E. Nichani, A. Damian, J. D. Lee, How transformers learn causal structure with gradient descent. *arXiv [Preprint]* (2024). <https://arxiv.org/abs/2402.14735> (Accessed 22 February 2024).
68. A. Saparov *et al.*, Testing the general deductive reasoning capacity of large language models using OOD examples. *Adv. Neural Inf. Process. Syst.* **36**, 3083–3105 (2024).
69. P. Reizinger *et al.*, Understanding LLMs requires more than statistical generalization. *arXiv [Preprint]* (2024). <https://arxiv.org/abs/2405.01964> (Accessed 3 May 2024).
70. X. Tang *et al.*, Large language models are in-context semantic reasoners rather than symbolic reasoners. *arXiv [Preprint]* (2023). <https://arxiv.org/abs/2305.14825> (Accessed 24 May 2023).
71. B. Wang, X. Yue, Y. Su, H. Sun, Grokked transformers are implicit reasoners: A mechanistic journey to the edge of generalization. *arXiv [Preprint]* (2024). <https://arxiv.org/abs/2405.15071> (Accessed 23 May 2024).
72. C. Anil *et al.*, Exploring length generalization in large language models. *Adv. Neural Inf. Process. Syst.* **35**, 38546–38556 (2022).
73. H. Zhou *et al.*, What algorithms can transformers learn? A study in length generalization. *arXiv [Preprint]* (2023). <https://arxiv.org/abs/2310.16028> (Accessed 24 October 2023).
74. N. Dziri *et al.*, Faith and fate: Limits of transformers on compositionality. *Adv. Neural Inf. Process. Syst.* **36**, 70293–70332 (2024).
75. K. Kudo *et al.*, Do deep neural networks capture compositionality in arithmetic reasoning? *arXiv [Preprint]* (2023). <https://arxiv.org/abs/2302.07866> (Accessed 15 February 2023).
76. M. Hahn, N. Goyal, A theory of emergent in-context learning as implicit structure induction. *arXiv [Preprint]* (2023). <https://arxiv.org/abs/2303.07971> (Accessed 14 March 2023).
77. E. Boix-Adserà *et al.*, "When can transformers reason with abstract symbols?" in *The Twelfth International Conference on Learning Representations* (2024).
78. Z. Xu, Z. Shi, Y. Liang, "Do large language models have compositional ability? An investigation into limitations and scalability" in *ICLR 2024 Workshop on Mathematical and Empirical Understanding of Foundation Models* (2024).
79. S. Arora, A. Goyal, A theory for emergence of complex skills in language models. *arXiv [Preprint]* (2023). <https://arxiv.org/abs/2307.15936> (Accessed 29 July 2023).
80. Z. Wang *et al.*, The buffer mechanism for multi-step information reasoning in language models. *arXiv [Preprint]* (2024). <https://arxiv.org/abs/2405.15302> (Accessed 15 October 2024).

81. X. Chen, D. Zou, What can transformer learn with varying depth? Case studies on sequence learning tasks. *arXiv [Preprint]* (2024). <https://arxiv.org/abs/2404.01601> (Accessed 2 April 2024).
82. S. Arora, Y. Li, Y. Liang, T. Ma, A. Risteski, Linear algebraic structure of word senses, with applications to polysemy. *Trans. Assoc. Comput. Linguist.* **6**, 483–495 (2018).
83. N. Elhage *et al.*, Toy models of superposition. *arXiv [Preprint]* (2022). <https://arxiv.org/abs/2209.10652> (Accessed 21 September 2022).
84. K. Park, Y. J. Choe, V. Veitch, The linear representation hypothesis and the geometry of large language models. *arXiv [Preprint]* (2023). <https://arxiv.org/abs/2311.03658> (Accessed 7 November 2023).
85. J. Pennington, R. Socher, C. Manning, "GloVe: Global vectors for word representation" in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, A. Moschitti, B. Pang, W. Daelemans, Eds. (Association for Computational Linguistics, Doha, Qatar) (2014), pp. 1532–1543.
86. A. Power, Y. Burda, H. Edwards, I. Babuschkin, V. Misra, Grokking: Generalization beyond overfitting on small algorithmic datasets. *arXiv [Preprint]* (2022). <https://arxiv.org/abs/2201.02177> (Accessed 6 January 2022).
87. N. Nanda, L. Chan, T. Lieberum, J. Smith, J. Steinhardt, Progress measures for grokking via mechanistic interpretability. *arXiv [Preprint]* (2023). <https://arxiv.org/abs/2301.05217> (Accessed 6 January 2022).
88. Z. Zhong, Z. Liu, M. Tegmark, J. Andreas, The clock and the pizza: Two stories in mechanistic explanation of neural networks. *Adv. Neural Inf. Process. Syst.* **36**, 27223–27250 (2024).
89. K. Lyu *et al.*, "Dichotomy of early and late phase implicit biases can provably induce grokking" in *The Twelfth International Conference on Learning Representations* (2023).
90. Z. Liu, E. J. Michaud, M. Tegmark, "Omnigrok: Grokking beyond algorithmic data" in *The Eleventh International Conference on Learning Representations* (2023).
91. N. Mallinar *et al.*, Emergence in non-neural models: Grokking modular arithmetic via average gradient outer product. *arXiv [Preprint]* (2024). <https://arxiv.org/abs/2407.20199> (Accessed 29 July 2024).
92. J. Song, Z. Xu, OOD Generalization via Composition. GitHub. <https://github.com/JiajunSong629/oood-generalization-via-composition>. Deposited 11 August 2024.