LLM Circuit Analyses Are Consistent Across Training and Scale

Curt Tigges [1] **Michael Hanna** [2] **Qinan Yu** [3] **Stella Biderman** [1]

Abstract

Most currently deployed large language models (LLMs) undergo continuous training or additional finetuning. By contrast, most research into LLMs' internal mechanisms focuses on models at one snapshot in time (the end of pre-training), raising the question of whether their results generalize to real-world settings. Existing studies of mechanisms over time focus on encoder-only or toy models, which differ significantly from most deployed models. In this study, we track how model mechanisms, operationalized as circuits, emerge and evolve across 300 billion tokens of training in decoder-only LLMs, in models ranging from 70 million to 2.8 billion parameters. We find that task abilities and the functional components that support them emerge consistently at similar token counts across scale. Moreover, although such components may be implemented by different attention heads over time, the overarching algorithm that they implement remains. Surprisingly, both these algorithms and the types of components involved therein can replicate across model scale. These results suggest that circuit analyses conducted on small models at the end of pre-training can provide insights that still apply after additional pre-training and over model scale.

1. Introduction

As LLMs' capabilities have grown, so has interest in characterizing their mechanisms. Recent work in mechanistic interpretability often seeks to do so via circuits: computational subgraphs that explain task-solving mechanisms (Wang et al., 2023; Hanna et al., 2023; Merullo et al., 2024; Lieberum et al., 2023). Circuits can be found and verified using a variety of methods (Conmy et al., 2023; Syed et al., 2023; Hanna et al., 2024; Kramár et al., 2024; Ferrando & Voita, 2024), with the aim of reverse-engineering models' task-solving algorithms.

Though much circuits research is motivated by LLMs' capabilities, the setting in which such research is performed often differs from that of currently deployed models. Crucially, while most LLM circuits work (Wang et al., 2023; Hanna et al., 2023; Merullo et al., 2024; Lieberum et al., 2023; Tigges et al., 2023) studies models at the end of pre-training, currently deployed models often undergo continuous training (OpenAI et al., 2024; Anthropic, 2024; Gemini Team et al., 2024) or are fine-tuned for specific tasks (Chung et al., 2022; Hu et al., 2021). Other subfields of interpretability have studied model development during training (Hu et al., 2023; Chang et al., 2023; Warstadt et al., 2020; Choshen et al., 2022; Chang & Bergen, 2022), but similar work on LLM mechanisms is scarce. Existing mechanistic work over training has studied syntactic attention structures and induction heads (Olsson et al., 2022; Chen et al., 2024; Singh et al., 2024), but has focused on small encoder or toy models. Prakash et al. (2024) examines circuits in 7-billion-parameter models post-finetuning, but the evolution of circuits during pre-training remains unexplored. This raises questions about whether circuit analyses will generalize if the model in question is further trained or fine-tuned.

We address this issue by exploring when and how circuits and their components emerge during training, and their consistency across training and different model scales. We study circuits in models from the Pythia suite (Biderman et al., 2023b) across 300 billion tokens, at scales from 70 million to 2.8 billion parameters. We supplement this with additional data from models ranging up to 12 billion parameters. Our results suggest remarkable consistency in circuits and their attributes across scale and training. We summarize our contributions as follows:

Performance acquisition and functional component emergence are similar across scale: Task ability acquisition rates tend to reach a maximum at similar token counts across different model sizes. Functional components like name mover heads, copy suppression heads, and successor heads also emerge consistently at similar points across scales, paralleling previous findings that induction heads emerge at roughly 2B-5B tokens across models of all scales

[1]The EleutherAI Institute [2]ILLC, University of Amsterdam [3]Brown University. Correspondence to: Curt Tigges <curt@curttigges.com>, Michael Hanna <m.w.hanna@uva.nl>, Qinan Yu <qinan_yu@brown.edu>, Stella Biderman <stella@eleuther.ai>.

Presented at the Mechanistic Interpretability Workshop at ICML 2024.

(Olsson et al., 2022).

Circuit algorithms can remain stable despite component-level fluctuations: Analysis of the indirect object identification (IOI; Wang et al., 2023) circuit across training and scale reveals that even when individual components change, the overall algorithm remains consistent, indicating a degree of algorithmic stability. The algorithm also tends to be similar for dramatically different model scales, suggesting that some currently-identified circuits may generalize, at least on simple tasks.

Taken as a whole, our results suggest that circuit analysis can generalize well over both (pre-)training and scale even in the face of component and circuit size changes, and that circuits studied at the end of training in smaller models can sometimes be informative for larger models as well as for models with longer training runs. We hope to see this validated for other circuits, especially more complex ones, confirming our initial findings.

2. Methods

2.1. Circuits

A **circuit** (Olah et al., 2020; Elhage et al., 2021; Wang et al., 2023) is the minimal computational subgraph of a model that is faithful to its behavior on a given task. At a high level, this means that circuits describe the components of a model—e.g., attention heads or multi-layer perceptrons (MLPs)—that the model uses to perform the task. A task, within the circuits framework, is defined by inputs, expected outputs, and a (continuous) metric that measures model performance on the task. For example, in the indirect object identification (IOI, (Wang et al., 2023)) task, the LM receives inputs like "When John and Mary went to the store, John gave a drink to", and is expected to output *Mary*, rather than *John*. We can measure the extent to which the LM fulfills our expectations by measuring the difference in logits assigned to *Mary* and *John*.

Circuits are useful objects of study because we can verify that are *faithful* to LM behavior on the given task. We say that a circuit is faithful if we can corrupt all nodes and edges outside the circuit without changing model behavior on the task. Concretely, we test faithfulness by running the model on normal input, while replacing the activations corresponding to edges outside our circuit, with activations from a corrupted input, which elicits very different model behavior. In the above case, our corrupted input could instead be "When John and Mary went to the store, Mary gave a drink to", eliciting *John* over *Mary*. If the circuit still predicts *Mary*, rather than *John*, it is faithful. As circuits are often small, including less than 5% of model edges, this faithfulness test corrupts most of the model, thus guaranteeing that circuits capture a small set of task-relevant model mechanisms. For more details on the circuits framework, see prior work and surveys (Conmy et al., 2023; Hanna et al., 2024; Ferrando et al., 2024).

Circuits have a number of advantages over other interpretability frameworks. As computational subgraphs of the model that flow from its inputs to its outputs, they provide complete explanations for a model's mechanisms. Moreover, their faithfulness, verified using a causal test, makes them more reliable explanations. This stands in contrast to probing (Belinkov, 2022), which only offers layer-representation-level explanations, and can be unfaithful, capturing features unused by the model (Elazar et al., 2020). Similarly, input attributions (Shrikumar et al., 2017; Sundararajan et al., 2017a) only address which input tokens are used, and may be unreliable (Adebayo et al., 2018; Bilodeau et al., 2024).

2.2. Circuit Finding

In order to find faithful circuits at scale over many checkpoints, we use efficient, attribution-based circuit finding methods. Such methods score the importance of all edges in a model's graph in a fixed number of forward and backward passes, independent of model size; though other patching-based circuit-finding methods (Conmy et al., 2023) are more accurate, they are too slow, requiring a number of forward passes that grows with model size. From the many existing attribution methods (Nanda, 2023; Ferrando & Voita, 2024; Kramár et al., 2024), we select edge attribution patching with integrated gradients (EAP-IG; Hanna et al., 2024) due to its faithful circuit-finding ability. Much like its predecessor, edge attribution patching (EAP; Nanda, 2023), EAP-IG assigns each edge an importance score using a gradient-based approximation of the change in loss that would occur if that edge were corrupted; however, EAP-IG yields more faithful circuits with fewer edges. Concretely, EAP-IG computes the score of an edge between nodes u and v, with activations z_u, z_v as

$$(z'_u - z_u)\frac{1}{m}\sum_{k=1}^{m}\frac{\partial L(z' + \frac{k}{m}(z - z'))}{\partial z_v}, \qquad (1)$$

where m is the number of integrated gradient steps (Sundararajan et al., 2017b) to perform. This method requires $\mathcal{O}(m)$ forward and backward passes to score all model edges; we choose $m = 5$ based on Hanna et al.'s (2024) recommendations.

After running EAP-IG to score each edge, we define our circuit by greedily searching for the edges with the highest absolute score. We search for the minimal circuit that achieves at least 80% of the whole model's performance on the task. We do this using binary search over circuit sizes; the initial search space ranges from 1 edge to 5% of the model's edges. The high faithfulness threshold we set

gives us confidence that our circuits capture most model mechanisms used on the given task. However, ensuring that a circuit is entirely complete, containing all relevant model nodes and edges, is challenging, and no definitive method of verifying this has emerged.

We use the circuits we identify through this method to identify key nodes and structures, but we do not limit our study of functional heads to components found through this method alone. Discussion of the size- and similarity-based metrics for these circuit graphs can be found in Appendix C.

2.3. Models

We study Biderman et al.'s (2023b) Pythia model suite, a collection of open-source autoregressive language models that includes intermediate training checkpoints. Though we could train our own language models or use another model suite with intermediate checkpoints (Sellam et al., 2022; Liu et al., 2023; Groeneveld et al., 2024), Pythia is particularly useful in providing a thorough set checkpoints for models at a variety of scales, all with identical training data. Each model in the Pythia suite has 154 checkpoints: 11 of these correspond to the model after 0, 1, 2, 4, ..., and 512 steps of training; the remaining 143 correspond to 1000, 2000, ..., and 143,000 steps. We find circuits at each of these checkpoints. As Pythia uses a uniform batch size of 2.1 million tokens, these models are trained on far more tokens (300 billion) than those in existing studies of model internals over time. We study models of varying sizes, from 70 million to 12 billion parameters.

2.4. Tasks

We examine the mechanisms behind four different tasks taken from the (mechanistic) interpretability literature. We choose simple tasks explicitly because they are feasible for even the smaller models we study to perform, and also because these tasks are simple enough that existing work has already provided clues and sometimes detailed descriptions of how models perform them. By contrast, we do not yet have circuit-level representations of more complex tasks and do not yet understand how models perform them. To verify that our models use similar circuits as heretofore-studied models to perform the simple tasks we selected we briefly analyze our models' indirect object identification and greater-than circuits in Appendix A. The other task are MLP-dominant and do not involve much attention head activity; for these circuits, we verify that this is still the case in Pythia models.

Indirect Object Identification The indirect object identification (IOI; Wang et al., 2023) task feeds models inputs such as "When John and Mary went to the store, John gave a drink to"; models should prefer *Mary* over *John*. Corrupted inputs, like "When John and Mary went to the store, Mary gave a drink to", reverse model preferences. We measure model behavior via the difference in logits assigned to the two names (*Mary* and *John*). We use a small dataset of 70 IOI examples created with Wang et al.'s (2023) generator, as larger datasets did not provide significantly better results in our experiments and this size fit into GPU memory more easily.

Gendered-Pronoun The Gendered-Pronoun task (Vig et al., 2020; Mathwin et al., 2023; Chintam et al., 2023) measures the gender of the pronouns that models produce to refer to a previously mentioned entity. Prior work has shown "So Paul is such a good cook, isn't"", models prefer the continuation "he" to "she"; we measure the degree to which this occurs via the difference in the pronouns' logits. In the corrupted case, we replace the "Paul" with "Mary"; we include opposite-bias examples as well. We craft 70 examples as in (Mathwin et al., 2023).

Greater-Than The Greater-Than task (Hanna et al., 2023) measures a model's ability to complete inputs such as $s =$"The war lasted from the year 1732 to the year 17" with a valid year (i.e. a year > 32). Task performance is measured via probability difference (prob diff); in this example, the prob diff is $\sum_{y=33}^{99} p(y|s) - \sum_{y=00}^{32} p(y|s)$. In corrupted inputs, the last two digits of the start year are replaced by "01", pushing the model to output early (invalid) years that decrease the prob diff. We create 200 Greater-Than examples with Hanna et al.'s (2023) generator.

Subject-Verb Agreement Subject-verb agreement (SVA), widely studied within the NLP interpretability literature (Linzen et al., 2016; Newman et al., 2021; Lasri et al., 2022), tasks models with predicting verb forms that match a sentence's subject. Given input such as "The keys on the cabinet", models must predict "are" over "is"; a corrupted input, "The key on the cabinet" pushes models toward the opposite response. We measure model performance via prob diff, taking the difference of probability assigned to verbs that agree with the subject, and those that do not. We use 200 synthetic SVA example sentences from (Newman et al., 2021).

3. Circuit Formation

3.1. Behavioral Evaluation

We begin our analysis of LLMs' task mechanisms over time by analyzing LLM behavior on these tasks; without understanding their task behaviors, we cannot understand their task mechanisms. We test these by running each model (Section 2.3) on each task (Section 2.4). Our results (Figure 1) display three trends across all tasks. First, all models but the weakest (Pythia-70m) tend to arrive at similar task

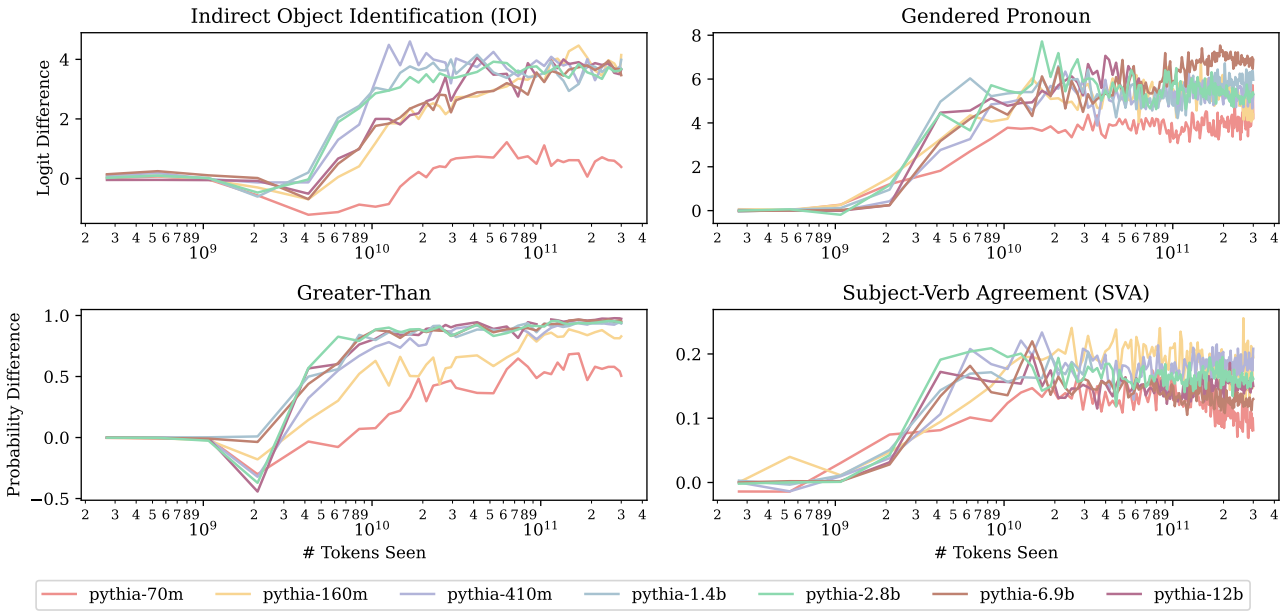


Figure 1. Task behavior across models and time (higher indicates a better match with expected behavior). Across tasks and scales, model abilities tend to develop at the same number of tokens. We use logit difference (the difference between the logits for the "correct" and "incorrect" names in the task) and probability difference (average probability for the correct and incorrect answer groups) as metrics, as these were used in the original works that examined these tasks. Often, models will show *negative* performance on tasks immediately prior to developing the ability to do them; we leave to future work why this is the case.

performance at the end of training. This is consistent with our choice of tasks: they are simple, learnable even by small models, and scaling does not significantly improve performance. Second, once models begin learning a task, their overall performance is generally non-decreasing, though there are minor fluctuations; Pythia-2.8b's logit difference on Gendered Pronouns dips slightly after it learns the task. In general, though, models tend not to undergo significant unlearning. The only marked downward trend (Pythia-70m at the end of SVA) comes from a weak model.

Finally, for each task we examined, we observed that there was a model size beyond which additional scale did not improve the rate of learning, and sometimes even decreased it; task acquisition appeared to approach an asymptote. We found this surprising due to the existence of findings showing the opposite trend for some tasks: (Kaplan et al., 2020; Rae et al., 2022). On some tasks (Gendered Pronouns and Greater-Than), all models above a certain size (70M parameters for Gendered Pronouns and 160M for Greater-Than) learn tasks at roughly the same rate. On IOI, models from 410M to 2.8B parameters learn the task the fastest, but larger models (6.9B and 12B) have learning curves more like Pythia-160m. We obtain similar results on more difficult tasks like SciQ (Welbl et al., 2017); results in Appendix F.

What drives this last trend, limiting how fast large models learn tasks? To understand this, we delve into the internal model components that support these behaviors and trends.

3.2. Component Emergence

Prior work (Olsson et al., 2022; Chen et al., 2024; Singh et al., 2024) has shown how a model's ability to perform a specific task can hinge on the development of certain components, i.e. the emergence of attention heads or MLPs with specific, task-beneficial behaviors. Prior work has also thoroughly characterized the components underlying model abilities in two of our tasks, IOI and Greater-Than, at the end of training. We thus ask: is it the development of these components that causes the task learning trends we saw before? We focus on four main components, all of which are attention heads, which we briefly describe here:

Induction Heads (Olsson et al., 2022) activate on sequences of the form `[A][B]...[A]`, attending to and upweighting `[B]`. This allow models to recreate patterns in their input, and supports IOI and Greater-Than.

Successor Heads (Gould et al., 2023) identify sequential values in the input (e.g. "11" or "Thursday") and upweight their successor (e.g. "12" or "Friday"); this supports Greater-Than behavior.

Copy Suppression Heads (McDougall et al., 2023) attend to previous words in the input, lowering the output probability of repeated tokens that are highly predicted in the

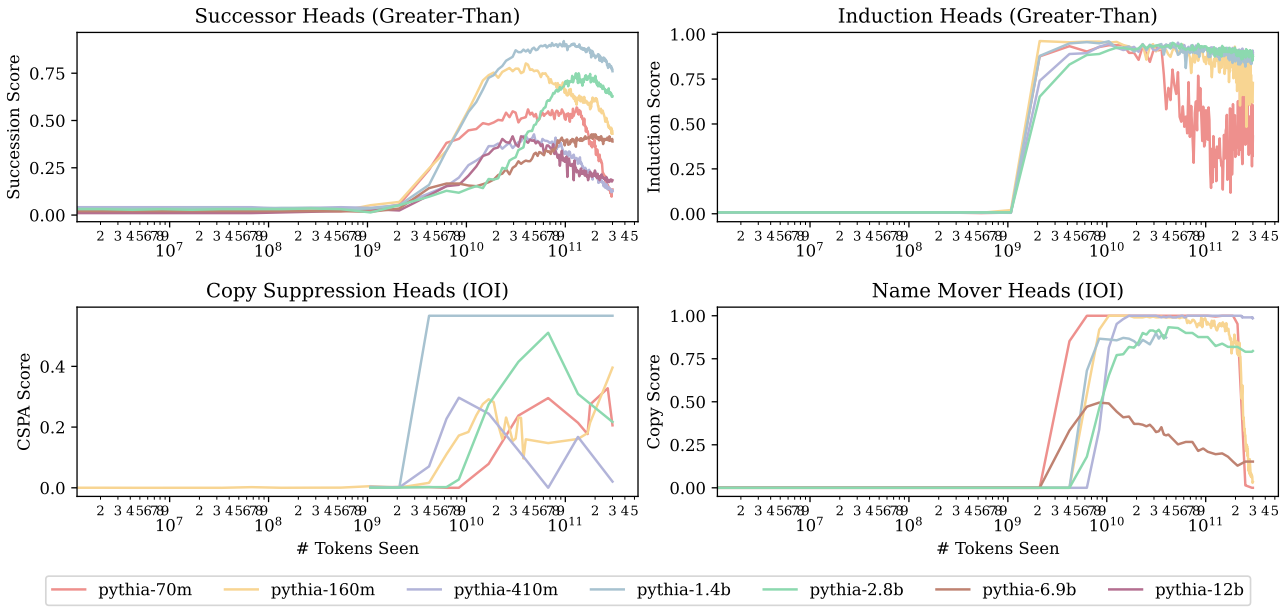


Figure 2. The development of components relevant to IOI and Greater-Than, across models and time. Each line indicates the strength of component behavior of the selected attention head from that model; higher values imply stronger component behavior. For each model and component, we plot the head in the relevant circuit (either IOI or Greater Than) that displays the component behavior the earliest.

residual stream input to the head. In the original IOI circuit, copy suppression heads hurt performance, downweighting the correct name. In contrast, we find (Appendix E) that they contribute positively to the Pythia IOI circuit by downweighting the incorrect name; this is possible because both names are already highly predicted in these heads' input, and they respond by downweighting the most repeated one.

Name-Mover Heads (Wang et al., 2023) perform the last step of the IOI task, by attending to and copying the correct name. Unlike the other heads described so far, this behavior is specific to IOI-type tasks; their behavior across the entire data distribution has not yet been characterized.

Because the importance of these components to IOI and Greater-Than has been established in other models, but not necessarily in those of the Pythia suite, we must first confirm their importance in these models. We do so by finding circuits for each model at each checkpoint using EAP-IG, as described in Section 2.2; we omit Pythia-6.9b and 12b from circuit finding for reasons of computational cost. We find that these component types indeed appear within the circuits of Pythia models' tasks circuits; see Appendix A and Appendix B for details on our methods and findings.

For each component, prior work has developed a metric to determine whether a model's attention head is acting like that component type; see Appendix E for details on these. Using these metrics, we score each of our models' heads for each of these behaviors at each checkpoint, evaluating the degree to which it acts like one of the four aforementioned heads. We then plot the earliest-emerging heads of each type, per model.

Our results (Figure 2) indicate that many of the hypothesized responsible components emerge the same time as model performance increases. Most models' induction heads emerge soon after they have seen 2×10^9 tokens, replicating the findings in (Olsson et al., 2022); immediately after this, Greater-Than behavior emerges. The successor heads, also involved in Greater-Than, emerge at a similar time.

For IOI, the name-mover heads emerge at similar timesteps (2 - 8×10^9 tokens) across models, with a very high strength, during or just before IOI behavior appears. Copy suppression heads emerge at the same timescale, but at varying speeds, and with varying strengths. Given that these heads are the main contributors to model performance in each task's circuit, and they emerge as or just before models' task performance increases, we can be reasonably sure that they are responsible for the emergence of performance. This said, we note an unusual trend: though model performance (Figure 1) does not decrease over time, the functional behavior of certain attention heads does. In the following section, we explain how this occurs.

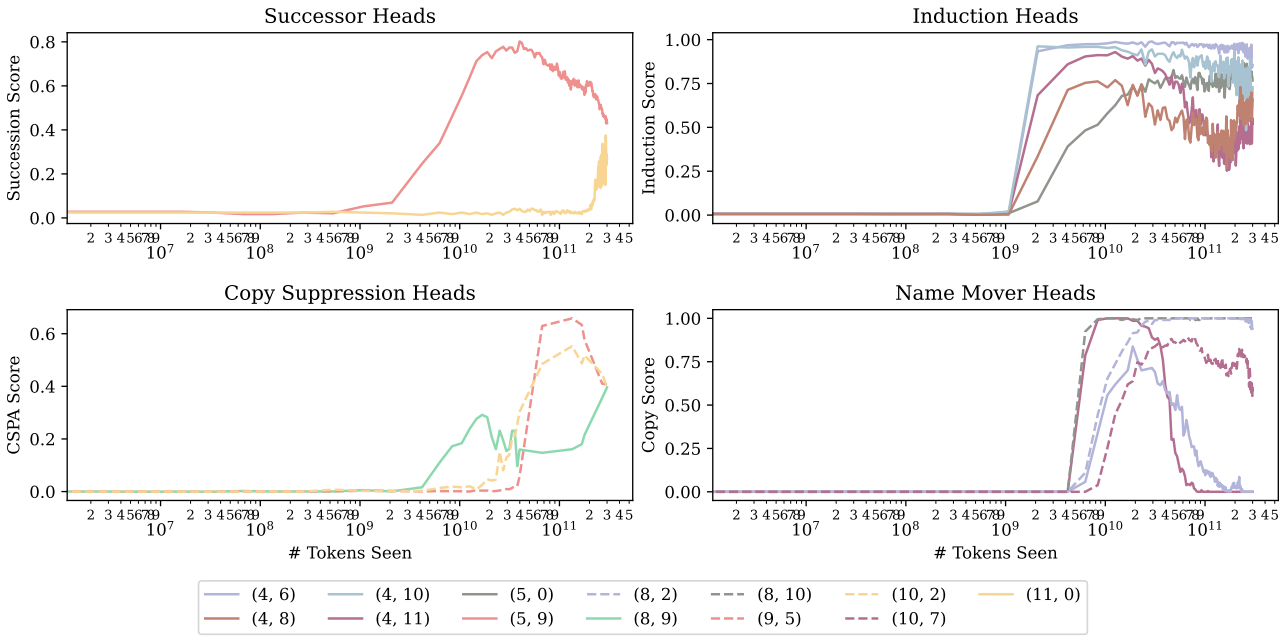


Figure 3. The development over time of various components relevant to IOI and Greater-Than in Pythia-160m. Here, we show the top heads for each function in the model. Each line indicates the degree to which an attention head, denoted as (layer, head), exhibits a given function; higher values imply stronger functional behavior. Heads often lose their current function; as this occurs, other heads take their place (but not always to the same degree or in the same numbers.

4. Algorithmic Stability and Generalizability in Post-Formation Circuits

We demonstrated in Section 3 that across a variety of tasks, models with differing sizes learn to perform the given task after the same amount of training; this appears to happen because each task relies on a set of components which develop after a similar count of training tokens across models. However, in Figure 2, we observed that attention heads that had a given function earlier in behavior can lose their function later in training. This raises questions: when the heads being used to solve a task change, does the algorithm implemented by the model change too? And how do these algorithms generalize across model scale?

4.1. Model Behavior and Circuit Components Post-Formation

To understand how model component behaviors change over time, we now zoom in on the components in one model, Pythia-160m, and study them over the course of training; where we earlier plotted only the top component (e.g. the top successor head), of each model, we now plot the top 5 of Pythia-160m's heads that exhibit a given functional behavior (or fewer, if fewer than 5 exist). By evaluating components and algorithms over Pythia-160m's 300B token training span, we go beyond previous work, which studies models trained on relatively few ($\leq$ 50M) tokens (Chen et al., 2024; Singh et al., 2024); in such work, components and task behaviors appear constant after component formation.

By contrast, our results (Figure 3) show that over the longer training period of Pythia models, the identity of components in each circuit is not constant. For example, the name-mover head (4,6) suddenly stops exhibiting this behavior at 3×10^{10} tokens, having acquired it after 4×10^{9} tokens. Similarly, Pythia-160m's main successor head (5,9) loses its successor behavior towards the end of training; however, (11,0) exhibits more successor behavior at precisely that time. Such balancing may lead to the model's task performance remaining stable, as we observed in the prior section (Figure 1). It seems plausible that self-repair (McGrath et al., 2023; Rushing & Nanda, 2024) contributes to this behavioral stability, but we leave the question of the exact "load-balancing" mechanism to future work. Nevertheless, models can clearly compensate for losses of and changes in individual circuit components.

4.2. Circuit Algorithm Stability Over Training

This instability of functional components raises an important question—when attention heads begin or cease to participate in a circuit, does the underlying algorithm change? To answer this, we examined the IOI circuit, as it is the most thoroughly characterized (Wang et al., 2023) circuit algorithm of our set of tasks. Our investigation follows a

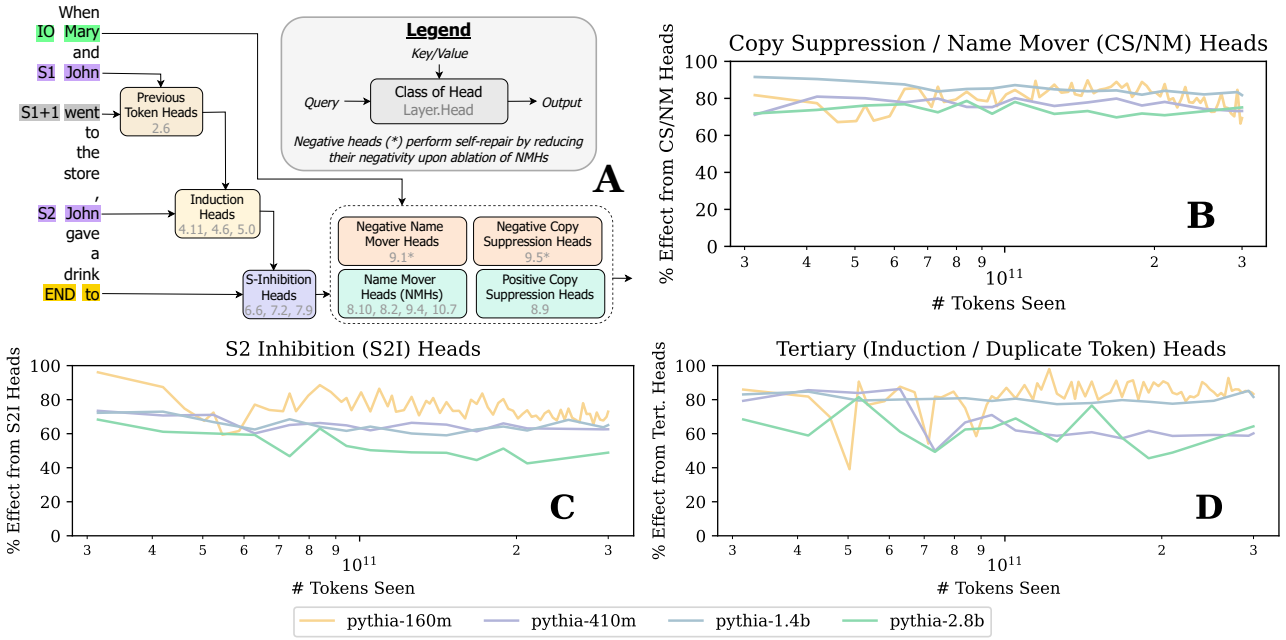


Figure 4. **A**: Pythia-160m's IOI circuit at the end of training (300B tokens). The remaining plots show the percent of model IOI performance that is explained by the Copy Suppression and Name-Mover Heads (**B**), the S-Inhibition Heads' edges to those heads (**C**), and the Induction / Duplicate Token Heads' connections to the S-Inhibition heads (**D**); higher percentages indicate that the corresponding edge is indeed important. Each of plots **B**-**D** verifies the importance of an edge from diagram **A**. The set of components analyzed changes from checkpoint to checkpoint such that all heads performing a relevant function (like name-moving) at that checkpoint are considered.

three-stage approach: first, we analyzed the IOI circuit at the end of training, reverse-engineering its algorithm; next, we developed a set of metrics to quantify whether the model was still performing that algorithm; finally, we applied these metrics across checkpoints, to determine if the algorithm was stable over training.

The first stage of our analysis is to analyze the IOI circuit at the end of training. Here, we present only the results of our analysis, but see Appendix B for details of this process, which follows the original analysis (Wang et al., 2023). Figure 4**A** shows the circuit that results from our analysis; it involves three logical "steps," each of which involves a different set of attention head types. Working backwards from the logit predictions, the direct contributors towards the logit difference are name-mover heads and copy suppression heads. The former attend to the indirect object in the prompt and copy it to the last position; the latter attend to and downweight tokens that appear earlier in the input. In the next step, the name-mover heads (but not the copy-suppression heads) use on token and positional information output by the S-inhibition heads to attend to the correct token. Finally, S-inhibition heads rely on information from induction heads and duplicate-token heads (only the former of which is involved in the IOI circuit for Pythia-160m in particular).

Next, we quantify the extent to which the circuit depends on each of these three steps, via path patching (Goldowsky-Dill et al., 2023), a form of ablation where activations are swapped with those from counterfactual prompts (see Appendix B for details). If a step is important, ablating the connection between the components involved in that step (e.g. in step 2, between induction / duplicate-token heads and S-inhibition heads) should have a large *direct effect*, and cause a large drop in model performance. For each step, our metric measures this direct effect, divided by the sum of the direct effects of ablating each edge with the same endpoint. Our metrics thus range from 0-100%; higher is better.

Finally, we compute each of these metrics for each model from 160M to 2.8B parameters in size.[1] We run them on each checkpoint post-circuit emergence (that is, when all component types appear in the circuit); for Pythia-160m, we test every checkpoint, and for the larger models we space out checkpoints to save compute, using approximately 1/3rd of the available checkpoints). We find (Figure 4**B**-**D**) that the behavior measured by these metrics is stable once the initial circuit has formed. Notably, in no model or metric are there dramatic shifts in algorithm corresponding to functional component shifts within the circuit. Moreover, all scores are

[1]We omit Pythia-70m, as it does not learn the task; due to computational constraints, we omit Pythia-6.9b/12b.

relatively high, generally above 50%; the core solvers of the algorithm, copy suppression and name-mover heads, have scores above 70%. This suggests that analyses of circuits in fully pre-trained models may generalize well to other model states, rather than being contingent on the particular checkpoint selected.

We emphasize that these metrics show algorithmic stability even in the face of component shifts; that is, many components of a particular type (e.g. name mover heads) can cease playing their role without perturbing the nature of the algorithm. Other heads start assuming the role of the components that have shifted away from their task, but this seems unlikely to be the only way the model can adapt to these kinds of changes. To further quantify the degree to which the set of component nodes involved in these circuits changes, we present a series of metrics in Appendix D.

Generalization across model scales also seems promising, as IOI circuit metrics from Pythia-160m are also high in larger Pythia variants. However, there is variation: while the name-mover, copy-suppression, and S-inhibition heads are at work in all models' circuits, the Pythia-160m circuit does not involve duplicate-token heads, while others do. So small differences exist amid big-picture similarity. Moreover, we stress that these algorithmic similarities might not hold for more complex tasks, for which a greater variety of algorithms could exist.

5. Discussion

Implications for Interpretability Research While our findings are based on a limited set of circuits, they hold significant implications for mechanistic interpretability research. Our study was motivated by the fact that most such research does not study models that vary over time, like currently deployed models. However, the stability of circuit algorithms over the course of training suggests that analyses performed on models at a given point during training may provide valuable insights into earlier and later phases of training as well. Moreover, the consistency in the emergence of critical components and the algorithmic structure of these circuits across different model scales suggests that studying smaller models can sometimes provide insights applicable to larger models. This dual stability across training and scale could reduce the computational burden of interpretability research and allow for more efficient study of model mechanisms. However, further research is needed to confirm these trends across a broader range of tasks and model architectures.

Limitations and Future Work Our analysis was limited to a narrow range of tasks feasible for small models. This limits in turn the scope of the claims that we can make. We believe it to be very possible that more complex tasks, not solvable by small models, which permit a larger range of algorithmic solutions, may show different trends from those that we discuss here. Such work would be valuable, though computationally expensive due to the model sizes required. Our analysis also studied models only from one model family, Pythia. It is thus not possible to tell if our results are limited to the specific model family we have chosen, which shares both architecture and training setup across model scale. Such work is in part hampered by the lack of large-scale model suites such as Pythia; future work could provide these suites to enable this sort of analysis.

Our work additionally only studies circuits over the course of training; in contrast, open-source models are more often fine-tuned, which could lead to different changes in mechanisms, though previous small-scale studies suggest this is not the case (Prakash et al., 2024). Finally, future work would do well to explore more complex phenomena, such as the self-repair and load-balancing mechanisms of LLMs, which ensure consistent task performance despite component fluctuations.

6. Related Work

Interpretability Over Time LLMs' development over the course of pre-training has been studied with various non-mechanistic interpretability techniques, particularly behavioral interpretability, which characterizes model behavior without making claims about its implementation. Such longitudinal analyses have studied LLM learning curves and shown that models of different sizes acquire capabilities in the same sequence (Xia et al., 2023; Chang et al., 2023), examined how LLMs learn linguistic information (Warstadt et al., 2020; Choshen et al., 2022; Chang & Bergen, 2022) and even predicted LLM behavior later in training (Hu et al., 2023; Biderman et al., 2023a). Nevertheless, behavioral studies alone cannot inform us about model internals. Prior work has studied the development of mechanisms in smaller models (Nanda et al., 2023; Olsson et al., 2022), and suggests that model mechanisms can change abruptly, even as models' outward behavior stays the same. Other previous studies have examined the pre-training window where acquisition of extrinsic grammatical capabilities occurs (Chen et al., 2024).

Mechanistic Interpretability We build on previous work in mechanistic interpretability, which aims to reverse engineer neural networks. *Circuits* are a significant paradigm of model analysis that has emerged from this field, originating with vision models (Olah et al., 2020) and continuing to transformer LMs (Meng et al., 2023; Wang et al., 2023; Hanna et al., 2023; Varma et al., 2023; Merullo et al., 2024; Lieberum et al., 2023; Tigges et al., 2023). Increasingly, research has tried to characterize the individual components

at work within circuits, not only at the level of attention heads (Olsson et al., 2022; Chen et al., 2024; Singh et al., 2024; Gould et al., 2023; McDougall et al., 2023), but also neurons (Vig et al., 2020; Finlayson et al., 2021; Sajjad et al., 2022; Gurnee et al., 2023; Voita et al., 2023) and other sorts of features (Bricken et al., 2023; Huben et al., 2024; Marks et al., 2024). Recent work has also tried to accelerate mechanistic research via automated techniques (Conmy et al., 2023; Bills et al., 2023; Syed et al., 2023; Hanna et al., 2024). Though mechanistic interpretability is a diverse field, it is often tied together by a reliance on causal methods (Vig et al., 2020; Chan et al., 2022; Geiger et al., 2021; 2023; Meng et al., 2023; Wang et al., 2023; Chan et al., 2023; Cohen et al., 2023), which provide more faithful mechanistic explanations.

Impact Statement

This paper aims to advance the field of Mechanistic Interpretability. By studying the stability and generalizability of language model circuits and components, our research contributes to understanding the degree to which mechanisms remain relevant across training and model scale. This understanding can aid in developing tools to detect and analyze critical behaviors in language models, in the long term potentially helping to identify and mitigate harmful or deceptive patterns in AI systems, thus enhancing their safety and reliability.

Contribution Statement

CT led and planned the study, wrote much of the experimental code, and performed many of the experiments. MH and QY assisted with experimental design, running experiments, and analyzing data. All three of the above worked together to discuss and decide upon research directions, write code, and write the manuscript. SB, as supervising author, provided critical feedback, helped shape the research, analysis, and manuscript, and supervised the project.

Acknowledgments

MH was funded in part by an OpenAI Superalignment fellowship. We would also like to thank Neel Nanda for feedback and advice on the project.

References

Adebayo, J., Gilmer, J., Muelly, M., Goodfellow, I. J., Hardt, M., and Kim, B. Sanity checks for saliency maps. In *Neural Information Processing Systems*, 2018. URL https://api.semanticscholar.org/CorpusID:52938797.

Anthropic. The claude 3 model family: Opus, sonnet, haiku, 2024. URL https://www-cdn.anthropic.com/de8ba9b01c9ab7cbabf5c33b80b7bbc618857627/Model_Card_Claude_3.pdf.

Belinkov, Y. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219, March 2022. doi: 10.1162/coli_a_00422. URL https://aclanthology.org/2022.cl-1.7.

Biderman, S., Prashanth, U. S., Sutawika, L., Schoelkopf, H., Anthony, Q. G., Purohit, S., and Raff, E. Emergent and predictable memorization in large language models. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023a. URL https://openreview.net/forum?id=Iq0DvhB4Kf.

Biderman, S., Schoelkopf, H., Anthony, Q., Bradley, H., O'Brien, K., Hallahan, E., Khan, M. A., Purohit, S., Prashanth, U. S., Raff, E., Skowron, A., Sutawika, L., and Van Der Wal, O. Pythia: a suite for analyzing large language models across training and scaling. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023b.

Bills, S., Cammarata, N., Mossing, D., Tillman, H., Gao, L., Goh, G., Sutskever, I., Leike, J., Wu, J., and Saunders, W. Language models can explain neurons in language models. https://openaipublic.blob.core.windows.net/neuron-explainer/paper/index.html, 2023.

Bilodeau, B., Jaques, N., Koh, P. W., and Kim, B. Impossibility theorems for feature attribution. *Proceedings of the National Academy of Sciences*, 121(2):e2304406120, 2024. doi: 10.1073/pnas.2304406120. URL https://www.pnas.org/doi/abs/10.1073/pnas.2304406120.

Bisk, Y., Zellers, R., Bras, R. L., Gao, J., and Choi, Y. Piqa: Reasoning about physical commonsense in natural language, 2019.

Bricken, T., Templeton, A., Batson, J., Chen, B., Jermyn, A., Conerly, T., Turner, N., Anil, C., Denison, C., Askell, A., Lasenby, R., Wu, Y., Kravec, S., Schiefer, N., Maxwell, T., Joseph, N., Hatfield-Dodds, Z., Tamkin, A., Nguyen, K., McLean, B., Burke, J. E., Hume, T., Carter, S., Henighan, T., and Olah, C. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023. https://transformer-circuits.pub/2023/monosemantic-features/index.html.

Chan, L., Garriga-Alonso, A., Goldwosky-Dill, N., Greenblatt, R., Nitishinskaya, J., Radhakrishnan, A., Shlegeris,

B., and Thomas, N. Causal scrubbing, a method for rigorously testing interpretability hypotheses. *AI Alignment Forum*, 2022. https://www.alignmentforum.org/posts/JvZhhzycHu2Yd57RN/causal-scrubbing-a-method-for-rigorously-testing.

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 [redwood research]. Alignment Forum, 2023. URL https://www.alignmentforum.org/posts/JvZhhzycHu2Yd57RN/causal-scrubbing-a-method-for-rigorously-testing. Accessed: 17th Sep 2023.

Chang, T. A. and Bergen, B. K. Word acquisition in neural language models. *Transactions of the Association for Computational Linguistics*, 10:1–16, 2022. doi: 10.1162/tacl_a_00444. URL https://aclanthology.org/2022.tacl-1.1.

Chang, T. A., Tu, Z., and Bergen, B. K. Characterizing learning curves during language model pre-training: Learning, forgetting, and stability, 2023.

Chen, A., Shwartz-Ziv, R., Cho, K., Leavitt, M. L., and Saphra, N. Sudden drops in the loss: Syntax acquisition, phase transitions, and simplicity bias in MLMs. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=MO5PiKHELW.

Chintam, A., Beloch, R., Zuidema, W., Hanna, M., and van der Wal, O. Identifying and adapting transformer-components responsible for gender bias in an English language model. In Belinkov, Y., Hao, S., Jumelet, J., Kim, N., McCarthy, A., and Mohebbi, H. (eds.), *Proceedings of the 6th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pp. 379–394, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.blackboxnlp-1.29. URL https://aclanthology.org/2023.blackboxnlp-1.29.

Choshen, L., Hacohen, G., Weinshall, D., and Abend, O. The grammar-learning trajectories of neural language models. In Muresan, S., Nakov, P., and Villavicencio, A. (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8281–8297, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.568. URL https://aclanthology.org/2022.acl-long.568.

Chung, H. W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., Li, Y., Wang, X., Dehghani, M., Brahma, S., Webson, A., Gu, S. S., Dai, Z., Suzgun, M., Chen, X., Chowdhery, A., Castro-Ros, A., Pellat, M., Robinson, K., Valter, D., Narang, S., Mishra, G., Yu, A., Zhao, V., Huang, Y., Dai, A., Yu, H., Petrov, S., Chi, E. H., Dean, J., Devlin, J., Roberts, A., Zhou, D., Le, Q. V., and Wei, J. Scaling instruction-finetuned language models, 2022.

Clark, P., Cowhey, I., Etzioni, O., Khot, T., Sabharwal, A., Schoenick, C., and Tafjord, O. Think you have solved question answering? try arc, the ai2 reasoning challenge, 2018.

Cohen, R., Biran, E., Yoran, O., Globerson, A., and Geva, M. Evaluating the ripple effects of knowledge editing in language models, 2023.

Conmy, A., Mavor-Parker, A. N., Lynch, A., Heimersheim, S., and Garriga-Alonso, A. Towards automated circuit discovery for mechanistic interpretability. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=89ia77nZ8u.

Elazar, Y., Ravfogel, S., Jacovi, A., and Goldberg, Y. Amnesic probing: Behavioral explanation with amnesic counterfactuals. *Transactions of the Association for Computational Linguistics*, 9:160–175, 2020. URL https://api.semanticscholar.org/CorpusID:227408471.

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.

Ferrando, J. and Voita, E. Information flow routes: Automatically interpreting language models at scale, 2024.

Ferrando, J., Sarti, G., Bisazza, A., and Costa-jussà, M. R. A primer on the inner workings of transformer-based language models, 2024.

Finlayson, M., Mueller, A., Gehrmann, S., Shieber, S., Linzen, T., and Belinkov, Y. Causal analysis of syntactic agreement mechanisms in neural language models. In Zong, C., Xia, F., Li, W., and Navigli, R. (eds.), *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 1828–1843, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.144. URL https://aclanthology.org/2021.acl-long.144.

Geiger, A., Lu, H., Icard, T., and Potts, C. Causal abstractions of neural networks. In *Advances in Neural Information Processing Systems*, volume 34, pp. 9574–9586, 2021. URL https://papers.nips.cc/paper/2021/hash/4f5c422f4d49a5a807eda27434231040-Abstract.html.

Geiger, A., Potts, C., and Icard, T. Causal abstraction for faithful model interpretation. Ms., Stanford University, 2023. URL https://arxiv.org/abs/2301.04709.

Gemini Team, Anil, R., Borgeaud, S., and et al., J.-B. A. Gemini: A family of highly capable multimodal models, 2024.

Goldowsky-Dill, N., MacLeod, C., Sato, L., and Arora, A. Localizing model behavior with path patching, 2023.

Gould, R., Ong, E., Ogden, G., and Conmy, A. Successor heads: Recurring, interpretable attention heads in the wild, 2023.

Groeneveld, D., Beltagy, I., Walsh, P., Bhagia, A., Kinney, R., Tafjord, O., Jha, A. H., Ivison, H., Magnusson, I., Wang, Y., Arora, S., Atkinson, D., Authur, R., Chandu, K. R., Cohan, A., Dumas, J., Elazar, Y., Gu, Y., Hessel, J., Khot, T., Merrill, W., Morrison, J., Muennighoff, N., Naik, A., Nam, C., Peters, M. E., Pyatkin, V., Ravichander, A., Schwenk, D., Shah, S., Smith, W., Strubell, E., Subramani, N., Wortsman, M., Dasigi, P., Lambert, N., Richardson, K., Zettlemoyer, L., Dodge, J., Lo, K., Soldaini, L., Smith, N. A., and Hajishirzi, H. Olmo: Accelerating the science of language models, 2024.

Gurnee, W., Nanda, N., Pauly, M., Harvey, K., Troitskii, D., and Bertsimas, D. Finding neurons in a haystack: Case studies with sparse probing, 2023.

Hanna, M., Liu, O., and Variengien, A. How does GPT-2 compute greater-than?: Interpreting mathematical abilities in a pre-trained language model. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=p4PckNQR8k.

Hanna, M., Pezzelle, S., and Belinkov, Y. Have faith in faithfulness: Going beyond circuit overlap when finding model mechanisms. In *First Conference on Language Modeling*, pp. To appear, October 2024. URL https://arxiv.org/abs/2403.17806.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. Lora: Low-rank adaptation of large language models, 2021.

Hu, M. Y., Chen, A., Saphra, N., and Cho, K. Latent state models of training dynamics. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=NE2xXWo0LF.

Huben, R., Cunningham, H., Smith, L. R., Ewart, A., and Sharkey, L. Sparse autoencoders find highly interpretable features in language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=F76bwRSLeK.

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., and Amodei, D. Scaling laws for neural language models, 2020.

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

Lasri, K., Pimentel, T., Lenci, A., Poibeau, T., and Cotterell, R. Probing for the usage of grammatical number. In Muresan, S., Nakov, P., and Villavicencio, A. (eds.), *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8818–8831, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.603. URL https://aclanthology.org/2022.acl-long.603.

Lieberum, T., Rahtz, M., Kramár, J., Nanda, N., Irving, G., Shah, R., and Mikulik, V. Does circuit analysis interpretability scale? evidence from multiple choice capabilities in chinchilla, 2023.

Linzen, T., Dupoux, E., and Goldberg, Y. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535, 2016. doi: 10.1162/tacl_a_00115. URL https://aclanthology.org/Q16-1037.

Liu, Z., Qiao, A., Neiswanger, W., Wang, H., Tan, B., Tao, T., Li, J., Wang, Y., Sun, S., Pangarkar, O., Fan, R., Gu, Y., Miller, V., Zhuang, Y., He, G., Li, H., Koto, F., Tang, L., Ranjan, N., Shen, Z., Ren, X., Iriondo, R., Mu, C., Hu, Z., Schulze, M., Nakov, P., Baldwin, T., and Xing, E. P. Llm360: Towards fully transparent open-source llms, 2023.

Marks, S., Rager, C., Michaud, E. J., Belinkov, Y., Bau, D., and Mueller, A. Sparse feature circuits: Discovering and editing interpretable causal graphs in language models, 2024.

Mathwin, C., Corlouer, G., Kran, E., Barez, F., and Nanda, N. Identifying a preliminary circuit for predicting gendered pronouns in gpt-2 small, 2023. URL https://itch.io/jam/mechint/rate/1889871.

McDougall, C., Conmy, A., Rushing, C., McGrath, T., and Nanda, N. Copy suppression: Comprehensively understanding an attention head, 2023.

McGrath, T., Rahtz, M., Kramar, J., Mikulik, V., and Legg, S. The hydra effect: Emergent self-repair in language model computations, 2023.

Meng, K., Bau, D., Andonian, A., and Belinkov, Y. Locating and editing factual associations in gpt, 2023.

Merullo, J., Eickhoff, C., and Pavlick, E. Circuit component reuse across tasks in transformer language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=fpoAYV6Wsk.

Nanda, N. Attribution Patching: Activation Patching At Industrial Scale, 2023. URL https://www.neelnanda.io/mechanistic-interpretability/attribution-patching.

Nanda, N., Chan, L., Lieberum, T., Smith, J., and Steinhardt, J. Progress measures for grokking via mechanistic interpretability. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=9XFSbDPmdW.

Newman, B., Ang, K.-S., Gong, J., and Hewitt, J. Refining targeted syntactic evaluation of language models. In Toutanova, K., Rumshisky, A., Zettlemoyer, L., Hakkani-Tur, D., Beltagy, I., Bethard, S., Cotterell, R., Chakraborty, T., and Zhou, Y. (eds.), *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 3710–3723, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.290. URL https://aclanthology.org/2021.naacl-main.290.

Nostalgebrist. interpreting GPT: the logit lens, 2020. URL https://www.lesswrong.com/posts/AcKRB8wDpdaN6v6ru/interpreting-gpt-the-logit-lens.

Olah, C., Cammarata, N., Schubert, L., Goh, G., Petrov, M., and Carter, S. Zoom in: An introduction to circuits. *Distill*, 2020. doi: 10.23915/distill.00024.001. https://distill.pub/2020/circuits/zoom-in.

Olsson, C., Elhage, N., Nanda, N., Joseph, N., DasSarma, N., Henighan, T., Mann, B., Askell, A., Bai, Y., Chen, A., Conerly, T., Drain, D., Ganguli, D., Hatfield-Dodds, Z., Hernandez, D., Johnston, S., Jones, A., Kernion, J., Lovitt, L., Ndousse, K., Amodei, D., Brown, T., Clark, J., Kaplan, J., McCandlish, S., and Olah, C. In-context learning and induction heads. *Transformer Circuits Thread*, 2022. https://transformer-circuits.pub/2022/in-context-learning-and-induction-heads/index.html.

OpenAI, Achiam, J., Adler, S., and et al., S. A. Gpt-4 technical report, 2024.

Prakash, N., Shaham, T. R., Haklay, T., Belinkov, Y., and Bau, D. Fine-tuning enhances existing mechanisms: A case study on entity tracking. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=8sKcAWOf2D.

Rae, J. W., Borgeaud, S., Cai, T., Millican, K., Hoffmann, J., Song, F., Aslanides, J., Henderson, S., Ring, R., Young, S., Rutherford, E., Hennigan, T., Menick, J., Cassirer, A., Powell, R., van den Driessche, G., Hendricks, L. A., Rauh, M., Huang, P.-S., Glaese, A., Welbl, J., Dathathri, S., Huang, S., Uesato, J., Mellor, J., Higgins, I., Creswell, A., McAleese, N., Wu, A., Elsen, E., Jayakumar, S., Buchatskaya, E., Budden, D., Sutherland, E., Simonyan, K., Paganini, M., Sifre, L., Martens, L., Li, X. L., Kuncoro, A., Nematzadeh, A., Gribovskaya, E., Donato, D., Lazaridou, A., Mensch, A., Lespiau, J.-B., Tsimpoukelli, M., Grigorev, N., Fritz, D., Sottiaux, T., Pajarskas, M., Pohlen, T., Gong, Z., Toyama, D., de Masson d'Autume, C., Li, Y., Terzi, T., Mikulik, V., Babuschkin, I., Clark, A., de Las Casas, D., Guy, A., Jones, C., Bradbury, J., Johnson, M., Hechtman, B., Weidinger, L., Gabriel, I., Isaac, W., Lockhart, E., Osindero, S., Rimell, L., Dyer, C., Vinyals, O., Ayoub, K., Stanway, J., Bennett, L., Hassabis, D., Kavukcuoglu, K., and Irving, G. Scaling language models: Methods, analysis & insights from training gopher, 2022.

Rushing, C. and Nanda, N. Explorations of self-repair in language models, 2024.

Sajjad, H., Durrani, N., and Dalvi, F. Neuron-level interpretation of deep nlp models: A survey, 2022.

Sakaguchi, K., Bras, R. L., Bhagavatula, C., and Choi, Y. Winogrande: An adversarial winograd schema challenge at scale, 2019.

Sellam, T., Yadlowsky, S., Tenney, I., Wei, J., Saphra, N., D'Amour, A., Linzen, T., Bastings, J., Turc, I. R., Eisenstein, J., Das, D., and Pavlick, E. The multiBERTs: BERT reproductions for robustness analysis. In *International Conference on Learning Representations*,

2022. URL https://openreview.net/forum?id=K0E_F0gFDgA.

Shrikumar, A., Greenside, P., and Kundaje, A. Learning important features through propagating activation differences. In Precup, D. and Teh, Y. W. (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 3145–3153. PMLR, 06–11 Aug 2017. URL https://proceedings.mlr.press/v70/shrikumar17a.html.

Singh, A. K., Moskovitz, T., Hill, F., Chan, S. C. Y., and Saxe, A. M. What needs to go right for an induction head? a mechanistic study of in-context learning circuits and their formation, 2024.

Sundararajan, M., Taly, A., and Yan, Q. Axiomatic attribution for deep networks. In *International Conference on Machine Learning*, 2017a. URL https://proceedings.mlr.press/v70/sundararajan17a/sundararajan17a.pdf.

Sundararajan, M., Taly, A., and Yan, Q. Axiomatic attribution for deep networks, 2017b. URL https://arxiv.org/abs/1703.01365.

Syed, A., Rager, C., and Conmy, A. Attribution patching outperforms automated circuit discovery, 2023.

Tigges, C., Hollinsworth, O. J., Geiger, A., and Nanda, N. Linear representations of sentiment in large language models, 2023.

Varma, V., Shah, R., Kenton, Z., Kramár, J., and Kumar, R. Explaining grokking through circuit efficiency, 2023.

Vig, J., Gehrmann, S., Belinkov, Y., Qian, S., Nevo, D., Sakenis, S., Huang, J., Singer, Y., and Shieber, S. Causal mediation analysis for interpreting neural nlp: The case of gender bias, 2020.

Voita, E., Ferrando, J., and Nalmpantis, C. Neurons in large language models: Dead, n-gram, positional, 2023.

Wang, K. R., Variengien, A., Conmy, A., Shlegeris, B., and Steinhardt, J. Interpretability in the wild: a circuit for indirect object identification in GPT-2 small. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=NpsVSN6o4ul.

Warstadt, A., Zhang, Y., Li, X., Liu, H., and Bowman, S. R. Learning which features matter: RoBERTa acquires a preference for linguistic generalizations (eventually). In Webber, B., Cohn, T., He, Y., and Liu, Y. (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 217–235, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.16. URL https://aclanthology.org/2020.emnlp-main.16.

Welbl, J., Liu, N. F., and Gardner, M. Crowdsourcing multiple choice science questions. In Derczynski, L., Xu, W., Ritter, A., and Baldwin, T. (eds.), *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pp. 94–106, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-4413. URL https://aclanthology.org/W17-4413.

Xia, M., Artetxe, M., Zhou, C., Lin, X. V., Pasunuru, R., Chen, D., Zettlemoyer, L., and Stoyanov, V. Training trajectories of language models across scales. In Rogers, A., Boyd-Graber, J., and Okazaki, N. (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13711–13738, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.767. URL https://aclanthology.org/2023.acl-long.767.

A. Analysis of Task Circuits

A.1. IOI Circuit & Algorithmic Criteria

To determine algorithmic consistency for the IOI circuit, we apply path patching as described in Appendix B in addition to using the component scores described in Appendix E. These are used to set thresholds for classifying attention heads. Though component score thresholds can be arbitrary, applying them consistently across all model checkpoints allows us to see the degree of similarity involved with model behavior.

Concretely, we use the following metrics and thresholds:

Direct-effect heads We initially perform path-patching on all model attention heads, measuring their impact on the logit different after the final layer of the model. We then classify attention heads as name-mover heads (NMHs), negative name-mover heads, and copy suppression heads (CSHs) based on copy score (for NMHs) or CPSA (for CSHs) of $> 10\%$, which yielded a small set of heads responsible for most of the direct effect. We measure the ratio of the absolute direct effect on logit difference for these heads vs. the total direct effect of all heads (including several unclassified heads) to obtain our first value.

Next, we conduct path-patching with NMHs as the receivers. This yields a set of heads that we then test for S2-inhibition (S2I) behavior, using Wang et al.'s (2023) test for the effect of token signal vs. positional signal: does the ablation of these positional signal heads A). reduce the logit difference through the NMHs, B). reduce NMH attention (which determines what they copy) to the indirect object token, and C). increase attention to the subject tokens? If a head meets all of these conditions, we classify it as an S2I head, as it emits a signal used by the NMHs to decide what to copy. The total absolute effect of these heads on the NMHs is then divided by the total absolute effect of all heads on the NMHs, producing our second measurement.

Finally, we conduct path-patching with S2I heads as receivers. Here, we apply a simpler test since these heads can be quite diffuse throughout the model: Do the heads involved have above-average induction or duplicate-token scores? If so, we classify them as induction heads or duplicate token heads (confirming via manual examination of attention patterns and behavior), and divide the total absolute effect of these heads by the total absolute effect of all heads on the S2I heads, producing our third measurement.

These three metrics capture the extent to which known and classifiable model components contribute at each of the three primary levels of the IOI circuit. If the degree to which unknown or unclassified components contribute to any part of the circuit, we will see the corresponding score drop. As we see that in practice they tend to stay level, we conclude that there is a high degree of stability for this circuit. Greater-Than Circuit & Algorithmic Criteria Hanna et al. (2023) identify three main properties of the Greater-Than circuit that we seek to quantify in our circuits. First, the MLPs with direct connections to the logits, especially the later of these MLPs upweight correct continuations. Second, the attention heads that connect to these MLPs upweight either the start year or the year after, allowing the MLPs to upweight the correct answers. Third, these attention heads attend to the start year position.

To quantify these, we perform the analysis performed by Hanna et al. (2023). At each checkpoint, we feed examples from our dataset into the model, and analyze the activations of the relevant attention heads and MLPs in the model using the logit lens (Nostalgebrist, 2020); that is, we multiply their output activations by the unembedding matrix, projecting them into vocabulary space. Then, considering the set of pseudo-logits corresponding to two-digit years (from 00 to 99), we verify the two aforementioned conditions. We measure for the MLPs, that the mean logit assigned to correct years is indeed higher than the mean logit assigned to incorrect years. For the attention heads, we verify that the year assigned the highest logit is either the start year of the sentence, or the following year. We then plot these. We also verify that they attend mainly to the start-year position.

B. Other Circuit-Analysis Methods

Circuit analysis can be conducted via a number of different methods; the method used to find the original IOI circuit (and that we use to verify algorithmic consistency in this task) is Wang et al.'s (2023) **path-patching**. Path patching is a specialized form of activation patching, used to isolate and analyze the influence of individual model components on a given task. Starting with two datasets (identical except for the key detail we want to base our circuit on, such as the correct and incorrect names in the IOI task), x_{orig} and x_{altered}, where x_{altered} is a counterfactual version of x_{orig}, the technique involves a sender attention head h and a set of receiver nodes $R \subseteq M$ within the model's computational graph M. Initially, activations

are recorded from both datasets. Subsequently, all attention heads except h are locked to their activations from x_{orig}, while h is updated with its activation from x_{altered}. This configuration allows for a forward pass on x_{orig}, capturing intermediate activations for nodes $r \in R$. A final forward pass on x_{orig} then patches R to these stored values, facilitating the assessment of h's impact on the model's output.

Path patching aims to gauge the significance of the path $h \to r$ by comparing the model's logit differences across multiple pairs $(x_{\text{orig}}, x_{\text{new}})$. By averaging these differences over many pairs, the method effectively measures the impact of specific paths on model performance, providing insights into the contributions of individual components to the overall task. The process is iterative, such that a practitioner would start by observing which nodes impact the logits directly, and then proceeding backwards to see what nodes affect those first direct-effect nodes, and so on.

C. Further Graph-Based Circuit Analyses

In the main body of the text, we examined circuits over the training process from component and algorithmic perspectives. But how do the circuit subgraphs themselves change over time and scale? In Section 2.2, we explain how these subgraphs are collected; we applied this method to Pythia-70m through Pythia-2.8b for the tasks listed in Section 2.4. The result is a set of nodes and edges for each model, checkpoint, and task. Here, we briefly examine some trends we identified in the analysis of this data.

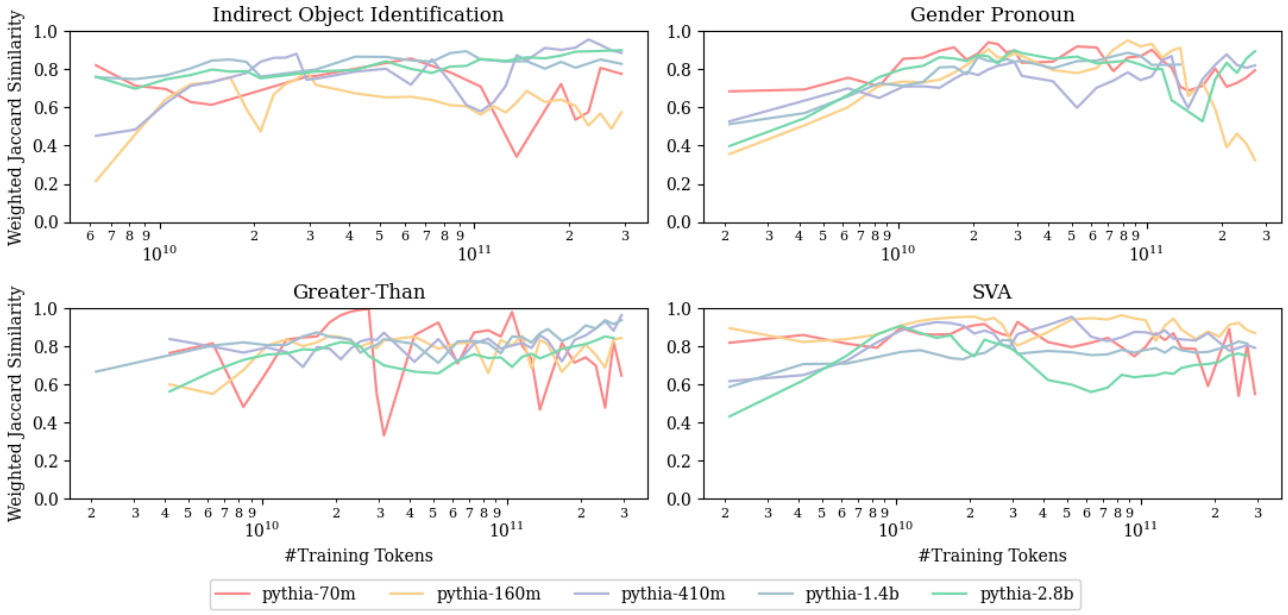


Figure 5. Exponentially-weighted moving average Jaccard similarity for circuit node sets over training token count. In general, larger models tend to have both higher average EWMA-JS and fewer abrupt fluctuations, indicating higher stability in the circuit constituents.

We first examine the consistency of the nodes in circuits over training. To measure this, we compute the Jaccard similarity (intersection over union) x_t between the circuits at each given checkpoint and those at all previous checkpoints. In order to smooth out local fluctuations and observe longer-term trends, we apply an exponential weighting with a decay factor $\alpha = 0.5$, such that the value at a given checkpoint is the exponentially-weighted Jaccard similarity with the complete set of previous checkpoints. We calculate the weighted Jaccard similarity $\hat{x_t}$ at checkpoint t: $\hat{x_t} = 0.5\hat{x_{t-1}} + 0.5x_t$, Our results (Figure 5) suggest that larger models tend to form more stable circuits (with both higher average values and fewer sharp fluctuations); EWMA-Jaccard is more volatile for Pythia-70m/160m. In the Gendered-Pronoun circuit, we observe that significant changes can occur even late in training.

We also compare the sizes (node counts) of the circuits over training. Across all four of our tasks, we find that the circuit size is positively correlated with the size of the models. We averaged node count across all the checkpoints for all the models on four tasks and calculated the pairwise correlation between the sizes of the models and the average node sizes. The Pearson correlation is r =0.72 for IOI and SVA, 0.9 for Greater-Than, 0.6 for Gender Pronoun. We also find a high degree

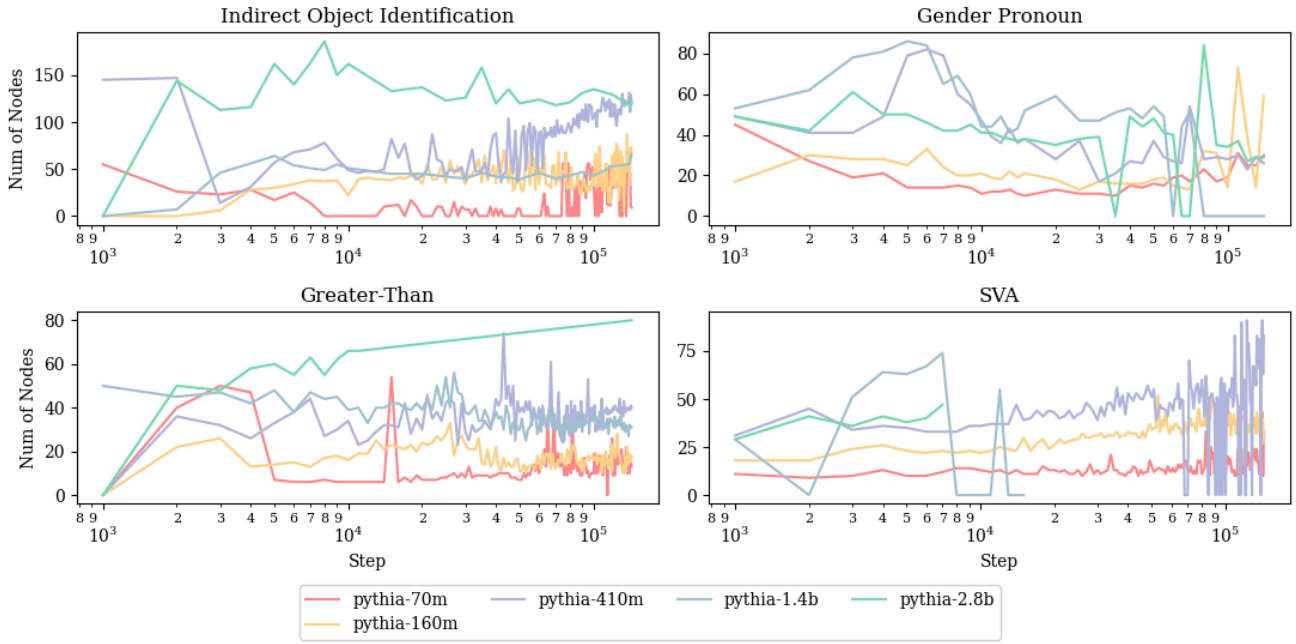


Figure 6. Number of Nodes in the circuits

of variability—circuit sizes can remain stable or fluctuate significantly, with no clear pattern based on the model or the task. We leave further exploration of why this is the case to future work, but present our size metrics in Appendix D, Figure 6.

D. Additional Size, Similarity & Change Rate Results

We graph out the number of nodes needed to generate faithful circuits across different checkpoints on all of the four tasks. Here we can observe that the number of nodes is positively correlated to the sizes of the models. When the model size increases the model needs more heads of the same kind to complete the same tasks. In the case of IOI, we can see the pink line for pythia-70m is at the bottom with the least number of nodes and the green line of 2.8b is at the top with the most number of nodes. This signifies a diffusion of the roles in attention heads. Increases in model size do not necessarily help heads become more specialized in their roles; rather, in these circuits more heads will take on the same roles.

E. Component Metrics

In this paper, we follow the metrics from previous literature in Wang et al. (2023) for name-mover heads, McDougall et al. (2023) for copy suppression heads, (Olsson et al., 2022) for induction heads, and (Gould et al., 2023) for successor heads.

Copy Score Following Wang et al. (2023), we check if the Name Mover Heads copy over the names across training time by using the same metrics- **copy score**. To validate the Name Mover Heads, we studied what values are written via the head's OV matrix. We take the state of the residual stream after the first layer of MLP on the specific name tokens. Then we multiply it with the OV matrix of the given heads, multiplied with the unembedding matrix and also the final layer norm. This simulates what will happen if the head attended perfectly to that token. We define copy score as the proportion of samples that contain the input name token in the top 5 logits.

CSPA Score McDougall et al. (2023) introduced a novel approach named copy suppression-preserving ablation (CSPA), designed to ablate all behaviors of a specified attention head except for those related to copy suppression. This method involves two distinct types of ablation: OV ablation and QK ablation. In the OV ablation process, the output of an attention head at a destination token D is represented as a weighted sum of result vectors from source tokens S, with the weights corresponding to the attention probabilities from D to S (Elhage et al., 2021). These vectors are then projected onto the unembedding vectors of their respective source tokens S, retaining only their negative components. Meanwhile, QK ablation

involves mean-ablating the result vectors from each source token S, except for the top 5% of source tokens that are most likely to be predicted at the destination token D based on the logit lens. For instance, in the phrase "All's fair in love and war," if the destination token D is "and" and the token "love" is a highly predicted follower of D and appears as a source token S, the result vector from S is projected onto the unembedding vector for "love," and everything else is mean-ablated. This demonstrates how the attention head in question suppresses the prediction of "love." To evaluate the impact of the ablation, the token distribution output by the model for a given prompt (π) is compared with the distribution following an ablation (π_{Abl}) using KL divergence $D_{KL}(\pi||\pi_{Abl})$. By averaging these values over the OpenWebText dataset, D_{CSPA} for CSPA and D_{MA} for a mean ablation baseline are obtained. The proportion of the effect explained is then calculated as $1 - \frac{D_{CSPA}}{D_{MA}}$, with KL divergence chosen because a value of 0 indicates that the ablated and clean distributions are identical, implying that 100% of the head's effect is explained by the preserved components.

Previous Token Score The Previous Token Score measures how effectively each attention head attends to the immediately preceding token. To compute this, we use a diagonal extraction on the attention pattern matrices, offset by one position. This captures the attention weights directed to the token that precedes each token in the sequence. The scores are averaged over all batches and tokens, providing a mean score for each attention head across all layers.

Duplicate Token Score The Duplicate Token Score evaluates the propensity of each attention head to focus on duplicate tokens within a sequence. We achieve this by creating input sequences where the original tokens are repeated consecutively. The attention pattern matrices are then examined for their focus on tokens that are exactly a sequence length apart, indicating duplicate attention. The scores are calculated by averaging the attention weights along the specified diagonal, representing the attention paid to duplicate tokens.

Induction Head Score Based on the prefix matching score described by Olsson et al. (2022), the Induction Head Score is designed to assess the ability of attention heads to engage in induction, where they predict the next token in a repeated sequence based on previously encountered patterns. To measure this, we generate sequences where a segment is repeated and compute the attention pattern matrices. We extract the diagonals offset by one less than the sequence length, capturing the attention from the end of the first segment to the start of the repeated segment. The mean attention scores along this diagonal provide the Induction Head Scores, averaged over all batches and tokens.

Succession Score The succession score (Gould et al., 2023) measures the degree to which an attention head performs succession, upweighting "2" in response to "1", or "May" given the input "April". As Gould et al.'s (2023) code is not publicly available, we re-implement their successor score as follows. We create a dataset of successor, consisting of numbers (in digit and written form), days of the week, and months. Then, we perform the following procedure from (Gould et al., 2023). Letting W_E and W_U denote the embedding and unembedding matrices of the model under study, MLP_0 denote the first (zero-indexed) MLP layer, and W_{OV} be the OV matrix of the head under study. Then $M = W_U W_{OV} MLP_0(W_E)$ is a square matrix whose size is that of the model vocabulary; each row thereof indicates, for the corresponding word x in the vocabulary, the degree to which an output word y is upweighted by the head under study, when x is in the input. For each (x, y) pair in our dataset (e.g. (3,4) or (Tuesday, Wednesday)) we verify that $M[x][y] > M[x][y']$ for all $y' \neq y$ in our dataset; that is, we ensure that the correct answer is more highly upweighted than any of the other possible answers in our dataset. The succession score is the proportion of examples in which that is the case.

F. Additional Evidence for Task-Dependent Learning Ceilings

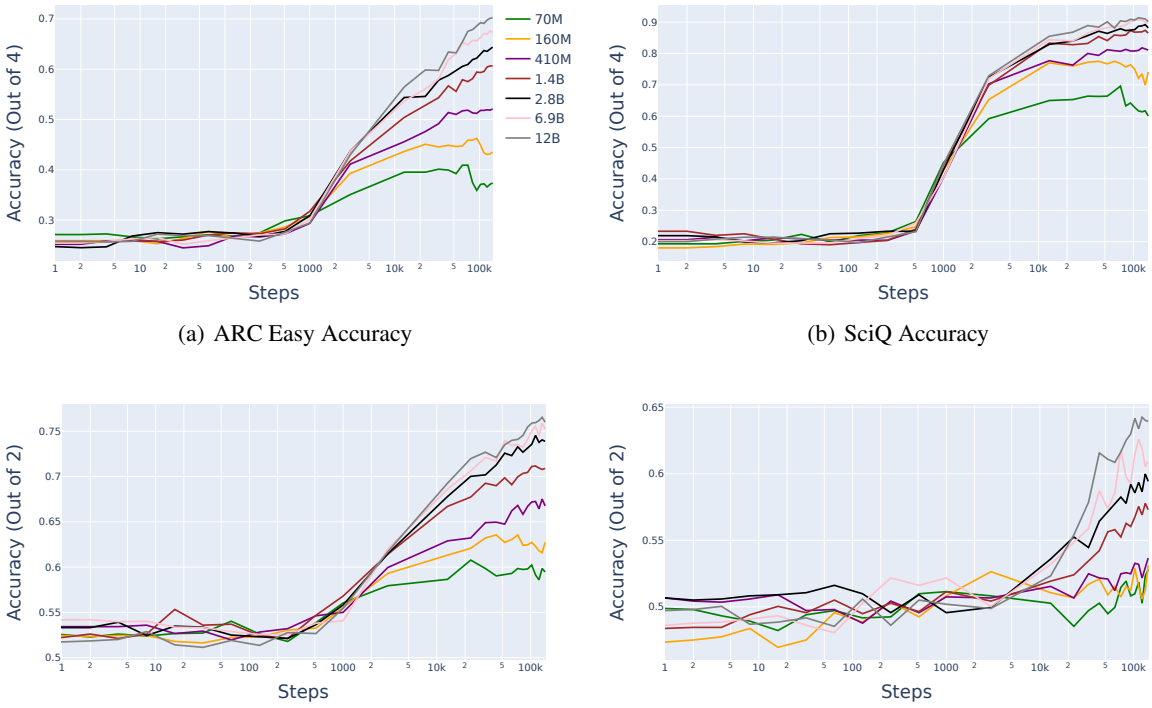
In addition to evaluations we performed ourselves, we also re-examined data collected during the Pythia training runs (Biderman et al., 2023b) on the SciQ (Welbl et al., 2017), PIQA (Bisk et al., 2019), WinoGrande (Sakaguchi et al., 2019), and ARC Easy (Clark et al., 2018) datasets. Each of these consist of a wide range of questions with multiple-choice answers, and accuracy was evaluated on the basis of the top choice logit produced by the model. We find that performance acquisition rates on these tasks followed the same pattern we detected with our simpler task datasets–that is, task learning rate seemed to approach an asymptote as the models increased in size. We describe the datasets below and present the results in Figure 7.

SciQ The Science Questions (SciQ) dataset (Welbl et al., 2017) consists of 13,679 crowdsourced multiple choice science exam questions ranging across physics, chemistry, biology, earth science, astronomy, and computer science. The questions cover a variety of complex reasoning skills such as causal reasoning, multi-hop inference, and understanding paragraph descriptions.

PIQA The Physical Interaction Question Answering (PIQA) dataset (Bisk et al., 2019) contains a total of 21k (across different subsets) multiple choice questions probing reasoning about basic physical commonsense knowledge. The questions test intuitive understanding of concepts like mass, volume, rigid objects, containment, stability, orientation, and more through grounded scenarios. Answering correctly requires applying physical reasoning.

ARC Easy The AI2 Reasoning Challenge (ARC) dataset (Clark et al., 2018) is a collection of 7,787 multiple choice science exam questions compiled from various grade-level sources, including a research partner of AI2. The questions cover diverse science topics and are structured as text-only prompts with 4 answer options. The ARC Easy subset consists of 5,197 of the relatively easier reasoning questions.

Winogrande The WinoGrande dataset (Sakaguchi et al., 2019) was inspired by the original Winograd Schema Challenge (WSC) and consists of 44k problems generated through crowdsourcing and systematic bias reduction algorithms. Most of these are relatively easy for humans, but often difficult for LLMs.



(a) ARC Easy Accuracy

(b) SciQ Accuracy

(c) PIQA Accuracy

(d) Winogrande Accuracy

Figure 7. Accuracy over training for four different datasets. Step numbers each represent approximately 2M tokens, so Step 1000 would be 2B tokens. We see that the rate of capability acquisition tends to approach an asymptote as models become larger.

G. Compute

Experiments were conducted over two months a pod of 8 A40 GPUs, each with 50 GB of GPU RAM. As an upper bound, our experiments would require all of these GPUs to operate for a month to run all of our experiments, but in practice we did not require all GPUs running simultaneously. We estimate that 0.25 utilization of this pod would be required in practice to run these experiments.

H. Licenses of Artifacts Used

The Pythia model suite is made available with an Apache 2.0 license. Wang et al.'s (2023) IOI dataset and Newman et al.'s (2021) SVA dataset are released under an MIT license. The remaining datasets (Greater-Than and Gendered-Pronouns) are

released without any license specified.