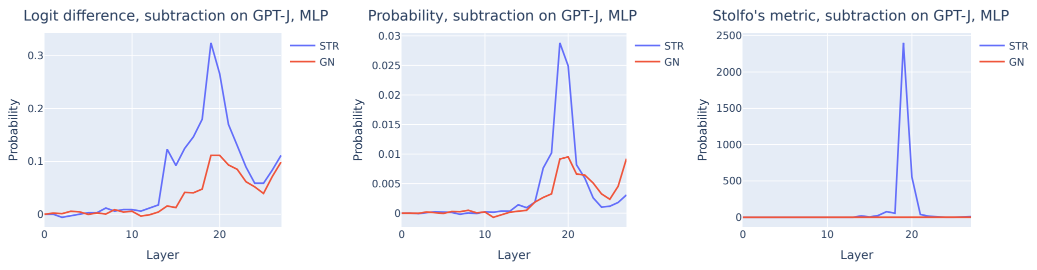
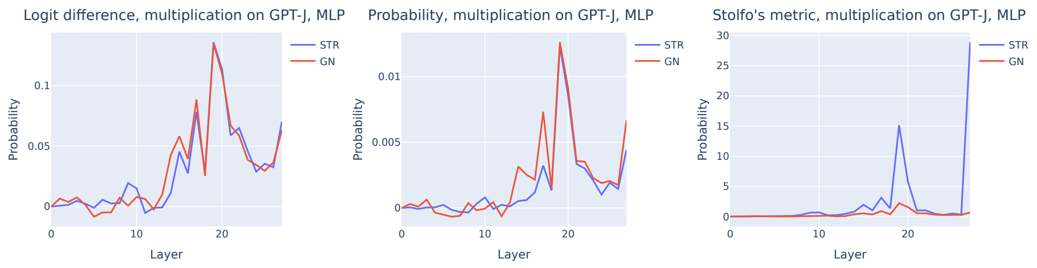


(a) Logit difference as the metric (b) Probability as the metric (c) Stolfo et al. (2023)'s metric

Figure 7: **The effects of patching MLP layers** in GPT-J on subtraction prompts.



(a) Logit difference as the metric (b) Probability as the metric (c) Stolfo et al. (2023)'s metric

Figure 8: **The effects of patching MLP layers** in GPT-J on multiplication prompts.

D RESULTS ON PYTHON DOCSTRING CIRCUIT

Heimersheim & Janiak (2023) studies a circuit for Python docstring completion in a pre-trained 4-layer attention-only Transformer.[5] We do not aim to fully replicate their results. Rather, we perform patching experiments to localize the important attention heads for the purpose of evaluating variants of activation patching.

Experimental setup In Heimersheim & Janiak (2023), a Python docstring completion instance consists of the following prompt:

```
def rand0(self, rand1, rand2, A_def, B_def, C_def, rand3):
    """rand4 rand5 rand6

    :param A_def: rand7 rand8
    :param B_def: rand9 rand10
    :param
```

where rand's are random single-token English words and the goal is to complete the prompt with C_def. Heimersheim & Janiak (2023) finds that the 4-layer model solves the docstring task with an accuracy of 56% and the logit difference is 0.5.

Following their approach, we run activation patching on all attention heads, across all token positions. This is more fine-grained than what we did for the IOI circuit, since the outcome would also highlight the token positions that matter for the important heads.

We apply corruption to the C_def token. For STR, it is replaced randomly by a single-token English word in the same way specified in Heimersheim & Janiak (2023).

Experimental results We take 200 instances of the docstring completion task, perform activation patching by positions and compute the patching effects. We report all position-head pairs with patching effect 2 standard deviations away from the mean. We find that the detections are mostly at the position of C_def and the last token of the prompt. The details are given in Table 2.

[5]The model is available in the TransformerLens library under the name attn-only-4l (Nanda & Bloom, 2022).

Corruption	Metric	At the position of `C_def`	At the last position
STR	Logit difference	0.0, 0.1, 0.5	2.3, 3.0, 3.5, 3.6
STR	Probability		3.0, 3.6
STR	KL divergence	0.0, 0.1, 0.5, 2.2	2.3, 3.0, 3.6
GN †	Logit difference	0.5	1.4, 1.5, 2.2, 2.3, 3.0, 3.6, 3.7
GN	Probability		3.0, 3.6
GN	KL divergence		2.2, 2.3, 3.0, 3.5, 3.6

Table 2: **Detections from activation patching** of attention heads by position on the Python docstring completion task. † Also detects two early-layer heads active at other positions and four negative heads active at the last position, which we omit here.

We again find that the localization outcomes are sensitive to the choice of corruption method and evaluation metric. The results of GN appear quite noisy, except when using probability as the metric. On the other hand, we remark that 3.0 and 3.6 are consistently highlighted across metrics and methods. In fact, they are typically assigned the largest patching effects (at the last position). This appears consistent with the result of Heimersheim & Janiak (2023), where 3.0 and 3.6 are found to be directly responsible for moving the `C_def` token.

E Results on the greater-than circuit in GPT-2 small

Hanna et al. (2023) In this section, we study the greater-tan task, specified below, and perform activation patching on the attention heads in GPT-2 small. In this setting, the prior work by Hanna et al. (2023); Conmy et al. (2023) show that model computation is fairly localized in this setting and provide a set of circuit discovery results. We remark that we do not attempt to replicate the circuit discovery results here, but rather to evaluate whether activation helps with localizing certain important model components.

Experimental setup Following Hanna et al. (2023), an instance of the greater-than task consists of an incomplete sentence of the template: "The `<noun>` lasted from the year XXYY to the year XX", where `<noun>` is a single-token word and XX and YY are two-digit numbers. For example, "The war lasted from year 1745 to 17". The goal is to complete the prompt with an integer greater than XX (in this case, 45). Across several metrics, Hanna et al. (2023) shows that GPT-2 small performs well on this task.

We focus on the role of attention heads in our study. To perform corruption, we ensure that the year XXYY are tokenized as [XX][YY] by filtering out years and numbers that do not conform to the constraint. GN corruption adds noise to the token embedding of YY. Following Hanna et al. (2023); Conmy et al. (2023), STR corruption replaces YY by 01. The probability metric, in this setting, is defined as the sum of probabilities of the years greater than YY. The logit difference metric is defined as the sum of logits of the years greater than YY minus the sum of logits of the years less than YY.

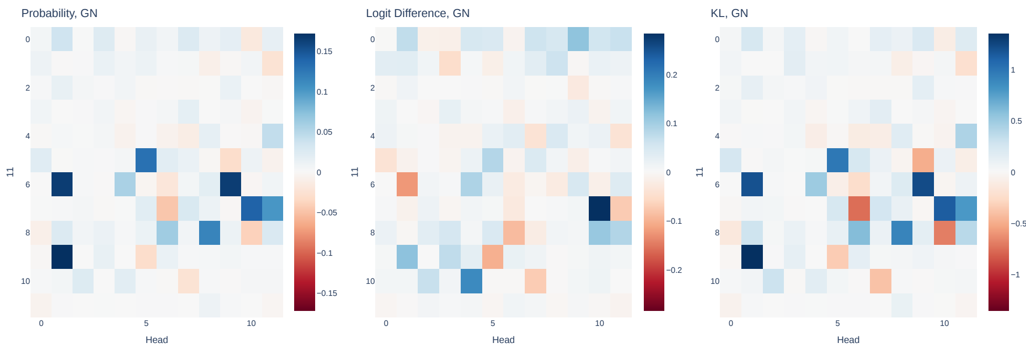
We perform activation patching on the attention heads outputs over all token positions.

Experimental results We find significant difference between the results achieved by GN and STR. In fact, the set of heads that are localized by the methods are mostly disjoint. Specifically, GN appears to give extremely noisy results that are not in line with the findings of Hanna et al. (2023); Conmy et al. (2023). The details are given in Table 3

The results from STR are fairly reasonable as the heads 6.9, 7.10, 8.11, 9.1 are also discovered by Hanna et al. (2023); Conmy et al. (2023), using more sophisticated methods. In contrast, the heads discovered by GN corruption share little overlap with STR, except 7.10 and 9.1. From visualizations, we also see that the plots for GN experiments are fairly noisy and do not yield much localization at all (Figure 9). On the other hand, the plots from STR are easily interpretable (Figure 10).

Corruption	**Metric**	Positive	Negative
STR	Logit difference	6.9, 7.10, 8.11, 9.1, 10.4	
STR	Probability	7.10, 8.11, 9.1	
STR	KL divergence	6.9, 7.10, 8.11, 9.1	10.7
GN	Logit difference	0.9, 7.10, 8.10, 9.1, 10.4	6.1, 8.6, 9.5
GN	Probability	5.5, 6.1, 6.9, 7.10, 7.11, 8.8, 9.1	
GN	KL divergence	5.5, 6.1, 6.9, 7.10, 7.11, 8.8, 9.1	5.9, 7.6

Table 3: **Detections from activation patching on attention heads for the greater-than task** in GPT-2 small, averaged across 300 prompts.



(a) Probability as the metric (b) Logit difference as the metric (c) KL divergence as the metric

Figure 9: **The effects of patching attention heads** in GPT-2 small on the greater-than task, using GN corruption. We see that the results are fairly noisy and do not appear to be localized.

F WHICH TOKENS TO CORRUPT MATTERS

In this section, we revisit the implementation of corruption methods in the setting of IOI (Wang et al., 2023).

Previously in our STR experiments in Section 3 and Section 4, the S2 token was corrupted by exchanging with IO. Similarly, in GN, we add noise to the token embedding of S2. We notice that the localization results from this approach miss at least 2 out of the 3 Name Mover (NM) Heads (Table 1); they directly contribute to the logit of IO as found by Wang et al. (2023). In particular, all combinations of metric and method would miss out on 9.6 and 10.0 (Table 5).

We show that by varying exactly which tokens to corrupt, the NMs can be discovered, too

Experimental setup We consider the IOI setting (Wang et al., 2023) using STR and GN corruption for localizing attention heads. Here, we corrupt the S1 and IO tokens. For STR, we simply replace S1 and IO by two random unrelated names. In both STR and GN experiments, the S2 token remains the same as in X_{clean}.

We perform activation patching across all attention heads. We apply logit difference, probability and KL divergence as the metric. All the results are averaged across 500 sampled IOI sentences.

Experimental results We find that most combinations of metrics and methods are able to notice all the NMs, when corruption applies to S1 and IO. We give the exact detections below and categorize them into positive and negative for simplicity.

First, we observe that this corruption seems to precisely target the NMs and their Negative counterparts. Intuitively, this is natural. Wang et al. (2023) finds that NMs write in the direction of the logit of the name (IO or S), whereas the Negative NMs do the opposite. Patching the clean activations of NMs recover such behavior.

Second, we confirm our finding that probability will miss out on the Negative NM; see Figure 11 for the plots.