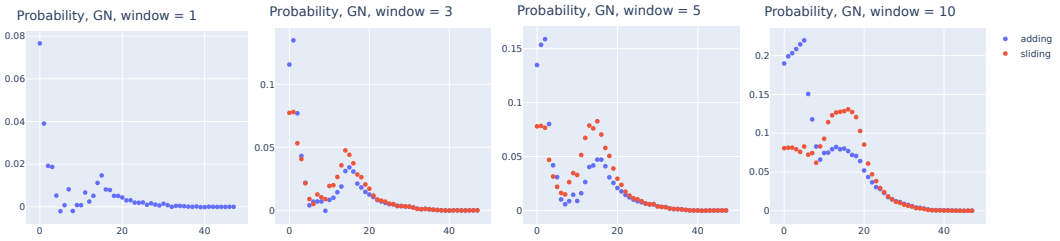

(a) Single-layer patching (b) Window size of 3 (c) Window size of 5 (d) Window size of 10

Figure 5: **Sliding window patching vs summing up individual patching effects**; patching MLP activation at the last subject token in GPT-2 XL on factual recall prompts. Sliding window patching offers $1.40\times$, $1.75\times$ and $1.59\times$ peak value than summation of single-layer patchings. Single-layer patching (a) suggests a weak peak.

under STR in probability is well within 2 SD from the mean (mean 0.003, SD 0.015, and 11.10 receives -0.022). Looking closely, the reason is simple:

- In the corrupted run of STR, the average probability of outputting the original IO is 0.03. Hence, the patching effect in probability, $\mathbb{P}_{\text{pt}}(\text{IO}) - \mathbb{P}_{*}(\text{IO})$, is at least -0.03, as $\mathbb{P}_{\text{pt}}(\text{IO})$ is non-negative. This is already close to 2 SD below the mean (-0.027). Hence, for an NNM to be detected via patching, its $\mathbb{P}_{\text{pt}}(\text{IO})$ needs to be near 0, which may be hard to reach.
- By contrast, under GN corruption, the average probability of IO is 0.13. Intuitively, this makes a lot more space for NNMs to demonstrate their effects.

In general, probability must fail to detect negative model components, if corruption reduces the correct token probability to near zero. We now give a cleaner experimental demonstration of this concern, using an original approach of Wang et al. (2023).

Experimental setup We revisit an alternative corruption method proposed by Wang et al. (2023), where S1, S2 and IO are replaced by three unrelated random names[2]; for example, "John and Mary [...], John" $\rightarrow$ "Alice and Bob [...], Carol." We use probability of the original IO as the metric. Intuitively, this replacement method would achieve much stronger corruption effect, since it removes all the relevant information (S and IO) of the original IOI sentence.

Experimental results First, we observe that the probability of outputting the IO of the original IOI sentence is negligible ($5\text{e}{-4}$) under this corruption. As a result, using probability detects neither NNMs. On the other hand, we find that logit difference still can. See Appendix H.3 for the plots. In Appendix F, we confirm the same finding when corruption is applied to S1 and IO only.

At a high-level, we believe that this is a pitfall of probability as an evaluation metric. Its non-negative nature makes it incapable of discovering negative model components in certain settings.

5 Sliding Window Patching

In this section, we examine the technique of sliding window patching in localizing factual information (Meng et al., 2022). For each layer, the method patches multiple adjacent layers simultaneously and computes the joint effects. Hence, one should interpret the result of Meng et al. (2022) as the effects being constrained within a window rather than at a single layer. We argue that such as hypothesis can be tested by an alternative approach and we compare the results from these two.

Experimental setup Instead of restoring multiple layers simultaneously, we patch each individual MLP layer one at a time. Then as an aggregation step, for each layer, sum up the single-layer patching effects of its adjacent layers. For example, we add up the effect at layer 2 to layer 6 to get an aggregated effect for layer 4. We patch the MLP output at the last subject token.

Experimental results For each window size, we compute the ratio of the maximum patching effect at the middle MLP layers between sliding window patching and summation of single-layer

[2]This corrupted distribution is denoted by p_{ABC} in the original paper of Wang et al. (2023)

patching. Over the combinations of window sizes, metrics and corruption methods, we find sliding window patching typically provides at least 20% more peak effect than the summation method.

In Figure 5, for window sizes of $3, 5, 10$, we plot the results using GN corruption and probability as the metric, the original setting as in Meng et al. (2022). We observe significant gaps between the sliding window and the summation method. Moreover, for single-layer patching, the peak at layer 15 is fairly weak (Figure 5a). Sliding window patching appears to generate more pronounced the concentration, as we increase the window sizes.

The result suggests that sliding window patching tends to amplify weak localization from single-layer patching (see Figure 12 for plots on single-layer MLP patching in GPT-2 XL). We believe this may arise due to certain non-linear effects in joint patching and therefore results from which should be carefully interpreted; see Section 6 for more discussions.

6 Discussion and Recommendations

We have observed a variety of gaps between corruption methods and evaluation metrics used in activation patching on language models. In this section, we summarize our findings and provide recommendations.

Corruption methods We are concerned that GN corruption puts the model off distribution by introducing noise never seen during training. Indeed, in Section 3.2, we provide evidence that in the corrupted run, model's internal functioning is OOD relative to the clean distribution. This may induce unexpected anomalies in the model behavior, interfering with our ability to localize behavior to specific components. Conceivably, GN corruption could even lead to unreliable or illusory results.

More broadly, this presents a challenge to any intervention techniques that introduce OOD inputs to the model or its internal layers, including ablations. In fact, similar concerns have been raised earlier in the interpretability literature on feature attribution as well; see e.g. Hooker et al. (2019); Janzing et al. (2020); Hase et al. (2021).

In contrast, STR provides counterfactual prompts ("The Eiffel Tower is in" vs "The Colosseum is in") that are in-distribution and thus induces in-distribution activations, avoiding the OOD issue. Therefore, we recommend STR whenever possible. GN may be considered as an alternative when token alignment or lack of analogous tokens makes STR unsuitable.

Evaluation metrics We generally recommend avoiding using probability as the metric, given that it may fail to detect negative model components.

We find logit difference a convincing metric for localization in language models. Consider an IOI setting where a model contains an attention head that boosts the logits of all (single-token) names. This head, though important, should not be viewed as part of the IOI circuit, but our interventions may still affect it.[3] By measuring $\text{Logit(IO)} - \text{Logit(S)}$, logit difference controls for such components and ensures they are not detected. This may not be achieved by other metrics, such as probability or Logit(IO) alone.

KL divergence tracks the full model output distributions, rather than focused only on the correct or incorrect answer, and can be a reasonable metric for circuit discovery as well (Conmy et al., 2023).

Sliding window patching We speculate that simultaneously patching multiple layers could capture the following non-linear effects and results in inflated localization plots:

- Joint patching may suppress the flow of corrupted information within the window of patched layers, where single-layer patching offers no such control.
- A window of patched layers may jointly perform a crucial piece of computation, such as a major boost to the logit of the correct token, which no individual layer can single-handedly achieve.

Generally, when examining the outcome from sliding window patching, one should be aware of the possibility of multiple layers working together. Thus, the results from the technique are to

[3] We note that if our interventions do not affect the head, then it will not show up on any metric.

be interpreted as the joint effects of the full window, rather than of a single layer. In practice, we recommend experimenting with single-layer patching first and only consider sliding window patching when individual layers seem to induce small effects.

Which tokens to corrupt? In some problem settings, a prompt contains multiple key tokens, all relevant to completing the task. This would offer the flexibility to choose which tokens to corrupt. This is another important dimension of activation patching. For instance, our experiments on IOI in Section 3 corrupt the S2 token. An alternative is to corrupt the S1 and IO. While this may seem an implementation detail, we find that this can greatly affect the localization outcomes.

Specifically, in Appendix F, we test corrupting S1 and IO in activation patching on IOI sentences, by changing their values to random names or adding noise to the token embeddings . We find that almost all techniques discover the 3 Name Mover (NM) Heads of the IOI circuit (Table 4 and Figure 11). These are attention heads that directly contribute to Logit(IO) as shown by Wang et al. (2023). In contrast, our prior experiments corrupting S2 miss most of them (Table 1).

We intuit that corrupting different tokens allows activation patching to trace different information within the model, thereby suggesting varying localizations results. For instance, in our prior experiments replacing S2 by IO, patching traces the value of IO or its position. On the other hand, in changing the values of S1 and IO while fixing their positions, patching highlights exactly where the model processes these values.

In practice, we recommend trying out different tokens to corrupt when the problem setting offers such flexibility. This may lead to more exhaustive circuit discovery.

7 Related work

Activation patching Activation patching is a variant of causal mediation analysis (Vig et al., 2020; Pearl, 2001), similar forms of which are used broadly in the interpretability literature (Soulos et al., 2020; Geiger et al., 2020; Finlayson et al., 2021; Geiger et al., 2022). The specific one with GN corruption was first proposed by Meng et al. (2022) under the name of causal tracing. Wang et al. (2023); Goldowsky-Dill et al. (2023) generalize this to a more sophisticated version of path patching.

Circuit analysis Circuit analysis provides post-hoc model interpretability (Casper et al., 2022). This line of work is inspired by Cammarata et al. (2020); Elhage et al. (2021). Other works include Geva et al. (2022); Li et al. (2023a); Nanda et al. (2023a); Chughtai et al. (2023); Zhong et al. (2023); Nanda et al. (2023b); Varma et al. (2023); Wen et al. (2023); Hanna et al. (2023); Lieberum et al. (2023). Circuit analysis often requires manual effort by researchers, motivating recent work to scale or automate parts of the workflow (Chan et al., 2022; Bills et al., 2023; Conmy et al., 2023; Geiger et al., 2023; Wu et al., 2023; Lepori et al., 2023).

Mechanistic interpretability (MI) MI aims to explain the internal computations and representations of a model. While circuit analysis is a major direction under this broad theme, other recent case studies of MI in language model include Mu & Andreas (2020); Geva et al. (2021); Yun et al. (2021); Olsson et al. (2022); Scherlis et al. (2022); Dai et al. (2022); Gurnee et al. (2023); Merullo et al. (2023); McGrath et al. (2023); Bansal et al. (2023); Dar et al. (2023); Li et al. (2023c); Brown et al. (2023); Katz & Belinkov (2023); Cunningham et al. (2023).

8 Conclusion

We examine the role of metrics and methods in activation patching in language models. We find that variations in these techniques could lead to different interpretability results. We provide several recommendations towards the best practice, including the use of STR as the corruption method.

In terms of limitations, our experiments are on decoder-only language models of size up to 6B. We leave it as a future direction to study other architectures and even larger models. Our work tests overriding corrupted activations by clean activations. The other direction—patching corrupted to clean—has also been used for circuit discovery, and it is interesting to compare these two. In addition, we provide tentative evidence that certain corruption methods lead to OOD model behaviors