LOST IN THE GAPS: HOW NON-LEXICAL FILLER DISRUPTS BELIEF TRACKING IN LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Unlike natural-language filler, repetitive, meaningless tokens such as a sequence of spaces can significantly degrade Language Models' (LMs) task accuracy when inserted into a prompt. Why does this happen? We investigate this question using a belief tracking task, in which a model must infer an agent's belief about the world, as a case study. Through activation patching and causal interventions on attention across three LMs, we find that filler sequences that are associated with performance degradation receive markedly high attention weights. Activation patching traces these failures to early MLP layers processing the filler: exchanging their outputs between harmful and harmless fillers suffices to remove or induce the errors. Patching individual MLP neurons further localizes these effects to a single neuron in Llama-3.1-8B and four neurons in Qwen2.5-14B, whereas in Qwen2.5-7B a few neurons suffice to remove the errors but inducing them requires a more distributed set. Our findings connect filler-induced failures to early MLP computations and downstream attention interactions, providing a mechanistic account of how semantically uninformative inputs disrupt belief tracking.

1 INTRODUCTION

Human readers pass over a long run of spaces or periods with little effort, whereas unrelated text is more likely to distract them. Language models may show the opposite sensitivity. A user who copies material from a web page or a PDF may unintentionally paste a long run of spaces into a prompt; to a human this seems harmless, yet it can substantially change the model's answer. Prior work has shown that longer inputs degrade performance even when the added content is only whitespace (Du et al., 2025). Understanding why such meaningless input disrupts language models therefore bears on their reliability in practical use. We refer to text that is inserted into a prompt and carries no task-relevant information as *filler*.

Irrelevant input is known to harm language models: irrelevant context distracts them (Shi et al., 2023), and the position of key information affects whether they use it (Liu et al., 2024). Du et al. (2025) attribute the harm caused by whitespace to input length itself. If length were the only cause, any filler of the same length should cause similar harm, and it remains unclear how length turns into errors inside the model. Models retrieve information from long inputs through a small set of specialized attention heads (Wu et al., 2025a; Lin et al., 2026), so any input that draws attention away from the task information could disrupt this retrieval. Tokens that carry little semantic information can attract a large share of attention (Xiao et al., 2024; Gu et al., 2025), and repeated tokens can trigger the same behavior (Yona et al., 2025). Such tokens are not limited to the start of the input: punctuation and other low-content tokens later in the input can also draw much attention, and reducing the attention they receive can improve accuracy (Yu et al., 2024). This suggests that the effect of length may depend on which tokens make up the filler: once repeated to a sufficient length, some tokens may attract a large share of attention and thereby disrupt how the model reads task information. It remains unclear, however, why some fillers are nearly harmless while others severely degrade performance, and which internal computations determine this difference. We study this question with belief tracking, a core component of theory of mind (ToM), the capacity to reason about mental states such as beliefs: a model reads a story and, based on the events a character observed, answers what that character believes a container holds (Prakash et al., 2026). This task suits our question because it requires the model to retrieve information bound in the story, its internal mechanisms are relatively well characterized (Feng & Steinhardt, 2024; Dai et al., 2024; Prakash et al., 2024; Gur-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Arieh et al., 2026), and ToM-related representations in language models are known to be sensitive to prompt variations and to depend on attention (Bortoletto et al., 2025; Wu et al., 2025b).

We insert filler between the story and the question and keep all other content fixed. Natural-language filler of equal length barely affects accuracy, whereas specific non-lexical tokens substantially degrade performance once they exceed a certain length. These fillers attract a large share of attention, and activation patching (Meng et al., 2022; Geiger et al., 2024; Heimersheim & Nanda, 2024) traces the harm to early MLP computations at the filler positions: exchanging these computations between harmful and harmless fillers suffices to remove or induce the errors.

In summary, we make three contributions:

1. We show that there is a clear difference between natural language tokens and non-lexical filler tokens in terms of when and how they disrupt processing (§3).

2. We show that much of the performance degradation results from the model placing disproportionate attention on the filler tokens, and that blocking attention to these tokens recovers much (but not all) of the performance (§3 and §4).

3. We localize the effect to early MLP computations at the filler positions: replacing a single early MLP output restores correct predictions or transfers errors, and a few of its neurons restore accuracy in all three models (§5).

2 Experimental Setup

We design our experiments to investigate why some repeated filler tokens reduce belief-tracking accuracy while others preserve it. We vary the filler while keeping task-relevant information fixed, then intervene on internal model computations to identify the components that contribute to these differences.

2.1 Task and Filler Conditions

We construct stories using the CausalToM template of Prakash et al. (2026). Each story describes two characters filling separate opaque containers, and the question asks what one character believes a specified container contains. We evaluate all three models on the same 100 story–question pairs, selected to have 100% no-filler accuracy in each model. Fifty questions concern the first character–container pair and fifty concern the second. We keep this evaluation set fixed across filler conditions.

Story: Lily and Amy are working in a busy restaurant. To complete an order, Lily grabs an opaque mug and fills it with port. Then Amy grabs another opaque pitcher and fills it with coffee.
[Filler: K **tokens]**
Question: What does Lily believe the mug contains?
Answer:

Figure 1: **A belief-tracking prompt with filler between the story and question.** Teal marks characters, orange marks containers, and pink marks their contents. The correct answer is `port`. The bracketed filler line marks the insertion site for K filler tokens; this label is not literal prompt text. We omit the fixed instruction for space; Appendix A gives the complete prompt.

Each prompt has the following structure (Figure 1):

`[Instruction] [Story] [Filler] [Question] [Answer]`

We compare seven filler types in two groups:

- **Repeated tokens:** *white space*, *dot*, *slash*, and '`hi`' repeat the space, period, slash, and space-prefixed `hi` token, respectively.
- **Text-based fillers:** *irrelevant text* is a passage with task-vocabulary words removed; *shuffled text* permutes its words; and *anagram* permutes letters within words while preserving word order and word lengths.

We vary the filler length K from 0 to 2,000 tokens. Across conditions, we keep the instruction, story, question, and correct answer unchanged. Appendix A gives the prompting details.

2.2 Models

We study three instruction-tuned language models: Llama-3.1-8B-Instruct (Grattafiori et al., 2024), and Qwen2.5-7B-Instruct and Qwen2.5-14B-Instruct (Yang et al., 2024). We obtain all three models from Hugging Face and run them in bfloat16. For attention analysis and interventions, we use 32-bit floating-point arithmetic for selected computations, including attention scores, softmax probabilities, and attention-weighted sums.

2.3 Evaluation

We compute accuracy by comparing the model's prediction from greedy decoding to the ground truth answer from a fixed set of 24 candidates listed in Appendix A. Each candidate corresponds to a single token when preceded by a space in the tokenizers of both model families. We compute accuracy as the proportion of examples for which the candidate with the highest logit matches the correct answer.

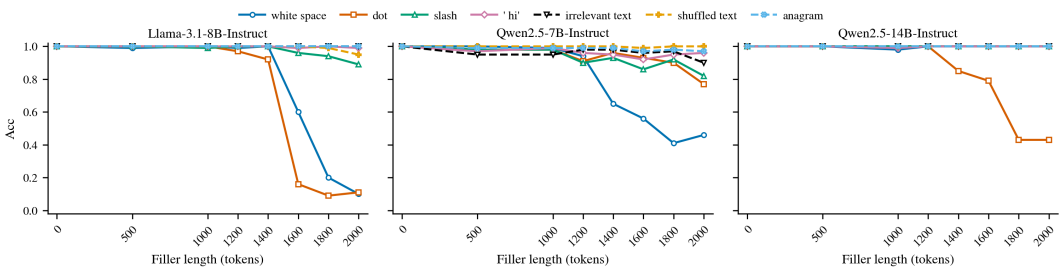

Figure 2: **Belief-tracking accuracy under different filler types and lengths.** We insert filler between the story and question. The x-axis counts inserted tokens; the y-axis shows the average accuracy on 100 examples. Solid lines repeat single tokens; dashed lines use text-based fillers. *Irrelevant text* refers to task-unrelated prose; *shuffled text* permutes its word order; *anagram* scrambles letters within words, preserving word order and lengths. We tokenize and truncate text fillers to the indicated length.

3 Attention Allocation to Filler Tokens

By design, the non-lexical fillers introduce no task-relevant information, yet only specific token types substantially reduce accuracy (Figure 2; Table 1). The same fillers also reduce accuracy on an unrelated factual question answer task (Appendix B).

We analyze attention from two kinds of positions, illustrated with the prompt in Figure 1: the *readout position*, the final token `Answer:`, whose next-token logits determine the prediction; and the question tokens, which include the queried character (`Lily`) and the queried container (`mug`). For layer ℓ and query position r (the readout position or a question token), let $a^{\ell}_{r,j}$ denote the attention weight from r to position j, averaged over attention heads, and let $\mathcal{F}$ denote the set of filler positions. We define the filler attention mass $A^{\ell}_{r} = \sum_{j \in \mathcal{F}} a^{\ell}_{r,j}$; for the question tokens, A^{ℓ}_{r} is averaged over those tokens. Unless stated otherwise, values are averaged over layers and the 100 examples.

Table 1: **Belief-tracking accuracy (%) without filler and with 2,000 filler tokens.** All filler rows use $K = 2000$; the first row uses $K = 0$. Filler definitions match Figure 2. Bold marks accuracy below 50%.

Filler	Llama-3.1-8B	Qwen2.5-7B	Qwen2.5-14B
No filler	100	100	100
White space	**10**	**46**	100
Dot	**11**	77	**43**
Slash	89	82	100
`' hi'`	99	96	100
Irrelevant text	100	90	100
Shuffled text	95	100	100
Anagram	100	97	100

We compare the four fillers that repeat a single token and call conditions with accuracy below 50% in Table 1 *degraded* and the remaining conditions *intact*.

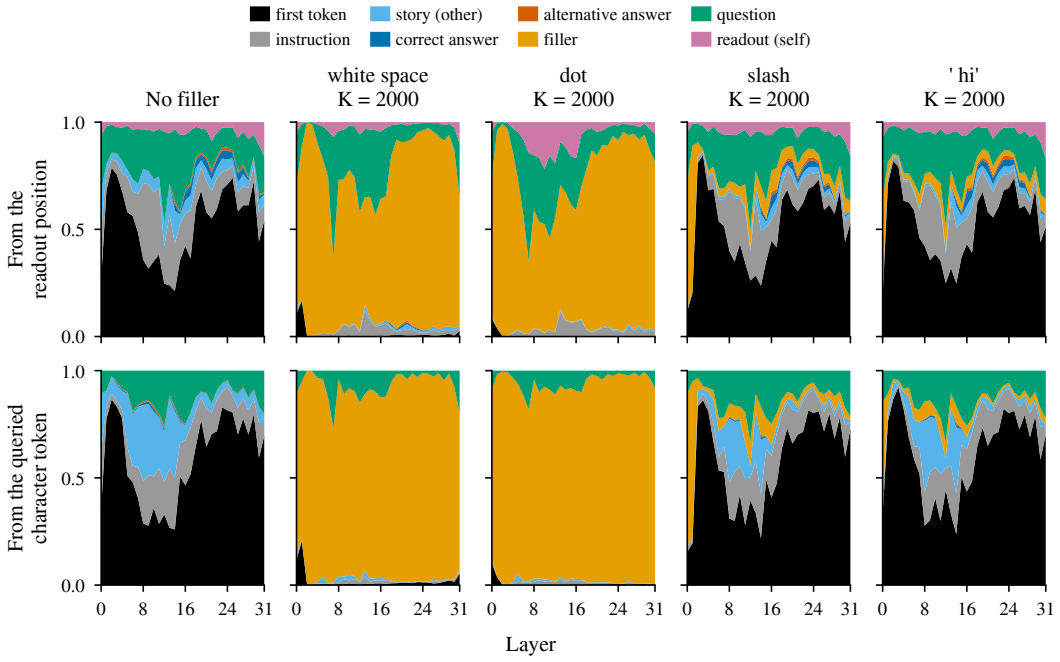

Figure 3: **Composition of attention in Llama-3.1-8B-Instruct.** Top: attention from the readout position. Bottom: attention from the queried character token in the question. Columns show the prompt without filler and the four fillers that repeat a single token, each with $K = 2000$. Colors partition the prompt into regions; in the example of Figure 1, *correct answer* is the story token `port` and *alternative answer* is `coffee`, the beverage in the other container.

In degraded conditions, the filler is the main target of attention at both query positions, whereas in intact conditions it receives little. In Llama-3.1-8B at $K = 2000$, dot filler receives 72.7% of readout attention and 85.4% of question-token attention (Table 5). Dot filler receives 77.2% of readout attention in Qwen2.5-14B, where it degrades accuracy, but 13.5% in Qwen2.5-7B, where white space filler instead degrades accuracy and receives the largest share (33.4%, compared with 14.2% in Qwen2.5-14B). The filler thus becomes the main target of attention in exactly the model–token combinations that are degraded. The filler dominates the attention that the first token and the story receive without filler (Figure 3). Without filler, the readout position and the question tokens attend mainly to the first token, which draws much of the attention (Xiao et al., 2024; Gu et al., 2025), and the queried character token also attends to the story. In Llama-3.1-8B with dot filler,

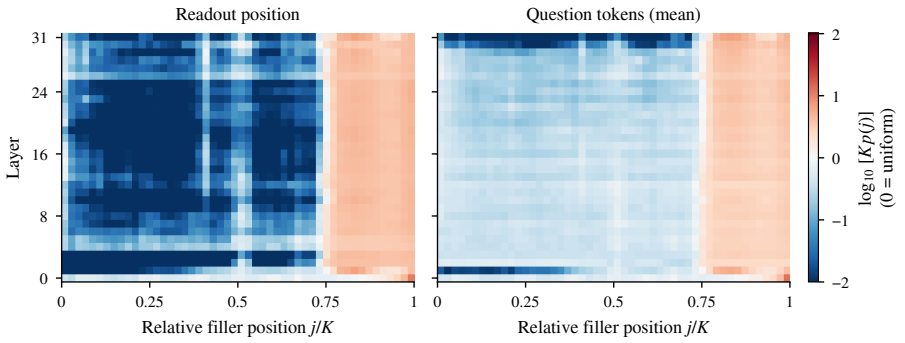

Figure 4: **Distribution of attention within dot filler in Llama-3.1-8B-Instruct** ($K = 2000$)**.** Left: readout position. Right: mean over question tokens. Color shows the attention to each filler position relative to a uniform distribution over the filler, on a logarithmic scale; red marks more than the uniform share. The horizontal axis gives the relative position within the filler, with 0 adjacent to the story, and the vertical axis gives the layer. Positions are grouped into 50 equal-width bins.

the filler absorbs most of the attention at both positions in all but the first two layers: the queried character token allocates 92.1% of its attention to the filler, and its attention to the story falls from 13.5% to 0.5%. At the readout position, the share of the first token falls from 52.2% to 0.6%, and attention to the story token naming the correct answer falls to 1% of its no-filler value. Slash and `'hi'` filler leave the allocation close to that without filler (Figures 11–13 for all models).

If a long filler merely diluted attention, attention would be spread roughly evenly across the filler. In degraded conditions, however, attention concentrates on positions in the latter part of the filler (Figure 4; Appendix C), and this concentration appears only when the filler is long, the range of lengths at which accuracy drops. The latter part of degraded filler thus takes over the role that the first token plays without filler: it carries no task-relevant information yet draws much of the attention.

In degraded conditions, the filler therefore becomes the main target of attention from both the readout position and the question tokens and concentrates this attention in its latter part, taking over the share held by the first token, and by the story, including the token naming the correct answer.

4 Interventions on Attention

Section 3 shows that, in degraded conditions, the filler becomes the main target of attention from post-filler positions. To test whether this attention participates in the accuracy loss, we remove it at inference time. We keep the input and position indices unchanged and, in every layer and attention head, set the attention scores from selected positions to the filler to $-\infty$ before the softmax. We compare three masking scopes: the *question mask* applies to the question tokens, the *readout mask* to the readout position, and the *both mask* to all positions after the filler. We evaluate the three masks in the four degraded conditions of Table 1.

Because the masks keep the position indices unchanged, masking all post-filler positions rules out an effect of the added distance between the story and the question (Figure 5). In both Qwen models, masking only the question tokens restores most of the accuracy, whereas masking only the readout position does not. The degradation in Qwen therefore arises largely from the question tokens attending to the filler, rather than from the readout position attending to it directly. In Llama, neither mask alone restores accuracy, suggesting that the filler affects the prediction through several post-filler positions at once.

Attention to the filler thus harms performance through two routes: the readout position attends to the filler directly, or the question tokens attend to it and pass the affected information on to the readout position. In Qwen, the harm comes mainly from the question tokens attending to the filler, whereas in Llama either route alone suffices to degrade accuracy.

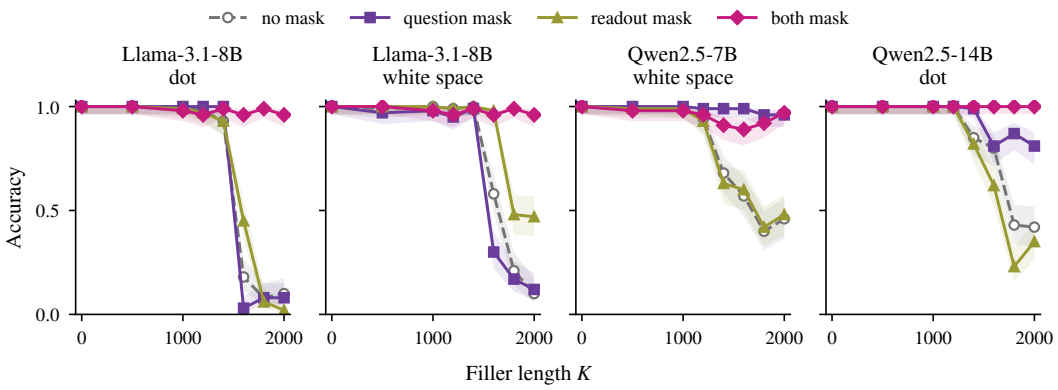

Figure 5: **Accuracy after masking attention from post-filler positions to the filler.** Each panel shows one degraded condition; the gray dashed line shows the unmasked model, and shaded bands show Wilson 95% confidence intervals. Unmasked accuracy is measured in the forward passes of this experiment.

Masking shows only whether the filler is needed for the harm, not how the filler causes it. To look inside this process, we replace the attention weights of the readout position in every layer with those it has under a different filler, while the content it reads still comes from the original prompt (Appendix F.2). Giving a harmless-filler prompt the attention weights of a harmful filler lowers accuracy to 1% in Llama-3.1-8B and 9% in Qwen2.5-14B, and the reverse replacement raises accuracy from 43% to 100% in Qwen2.5-14B; it restores accuracy only partly in Llama-3.1-8B, and both effects are small in Qwen2.5-7B. The filler therefore causes errors not through its content but by changing where the model directs its attention.

5 Causal Localization of Filler-Induced Disruption

We next locate the internal computations that give rise to the accuracy loss. The loss arises from how the model processes the filler itself rather than from story information carried into the filler: when we replace the internal states at the filler positions with those computed for the same filler placed after an unrelated passage of equal length, the accuracy loss persists (Appendix E). We therefore exchange activations at the filler positions between a harmful filler and a harmless filler of equal length: replacing the harmful activations tests whether the errors disappear, and the reverse replacement tests whether they appear. We apply this exchange first to attention and MLP modules and then to individual MLP neurons.

5.1 Module-Level Localization

Module-level interventions replace the output of one attention module or MLP at the filler positions with the output of the same module under the other filler condition, and all later computations proceed from the modified residual stream (Appendix G). Because the two conditions differ only in the filler token and only the filler positions are modified, any resulting change in the prediction must reach it through attention from post-filler positions to the filler.

Repair replaces the outputs of the degraded condition with those of the intact condition, and *damage* applies the reverse replacement. For each model, we pair a degraded filler with the intact filler of highest accuracy, which maximizes the accuracy gap between the two conditions: dot with `'hi'` in Llama-3.1-8B, white space with `'hi'` in Qwen2.5-7B, and dot with white space in Qwen2.5-14B, all at $K = 2000$. We locate effective layers on 50 localization examples and test them on the remaining 50 held-out examples.

Effective replacements concentrate in the earliest layers (Figure 6a–c). In Llama-3.1-8B, repair at layer 0 raises accuracy from 10% to 92% through the attention output and to 96% through the MLP output, whereas no replacement from layer 2 onward changes accuracy; in both Qwen models, repair

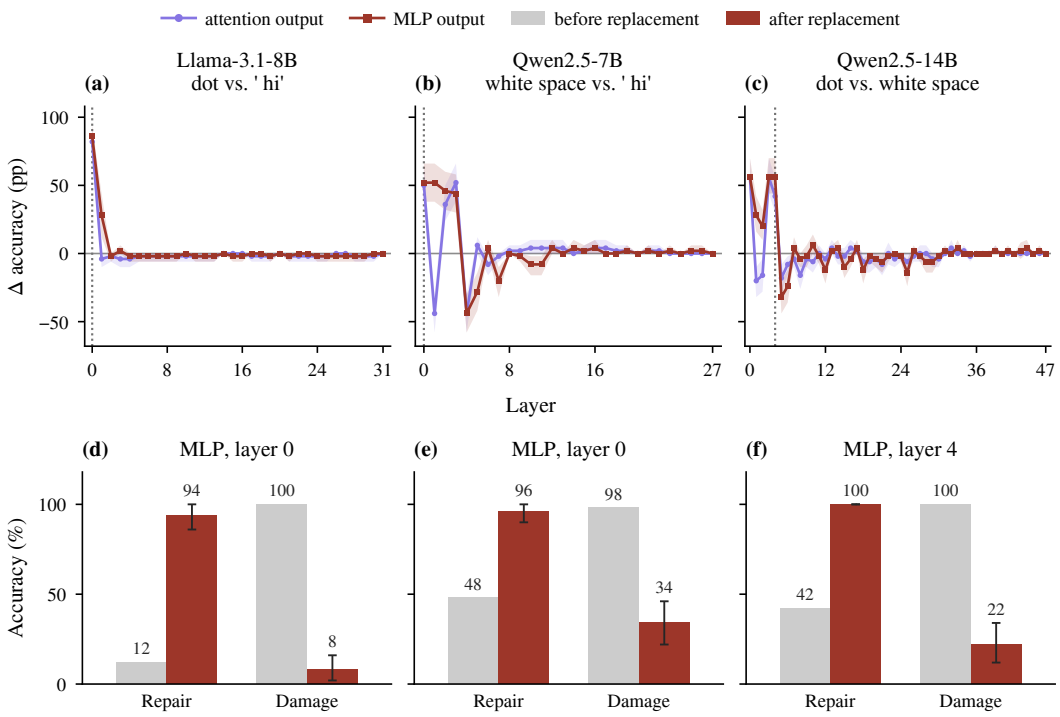

Figure 6: **Module-level interventions at the filler positions.** (a–c) Repair on the localization examples: change in accuracy of the degraded condition, in percentage points, after replacing the attention or MLP output of each layer; dotted lines mark the MLP layer tested in (d–f). (d–f) Accuracy before and after repair and damage at that layer on the held-out examples. Shaded bands in (a–c) show pointwise 95% paired bootstrap intervals of the accuracy change, and error bars in (d–f) show 95% bootstrap intervals of the accuracy after the replacement (2,000 resamples of examples).

likewise succeeds only within layers 0–4. In the damage direction, single-layer MLP outputs induce errors in all three models, whereas attention outputs do so only in Qwen2.5-7B (Appendix G).

We then fix, for each model, the MLP layer with the strongest effects in both directions (layers 0, 0, and 4) and repeat both replacements on the held-out examples (Figure 6d–f). In Llama-3.1-8B, repair raises accuracy on the dot condition from 12% to 94%, and damage lowers accuracy on the `'hi'` condition from 100% to 8%; both Qwen models show the same bidirectional effect. The replacements also shift the attention identified in Section 3: in Llama-3.1-8B, repair lowers the readout position's attention to the filler to the level of the intact condition, and damage raises it to the level of the degraded condition (Table 15). In both Qwen models, replacing only the keys at the filler positions likewise transfers the effect in both directions (Appendix D). Exchanging a single early MLP output at the filler positions thus suffices to restore correct predictions or to transfer errors.

5.2 Neuron-Level Localization

We next ask whether a few neurons of the fixed MLP reproduce these effects. We call each component of the MLP's intermediate activation vector, the input to its output projection, a neuron, and replace only the selected neurons at every filler position, leaving everything else unchanged (Appendix H). On the localization examples, we rank neurons by how much their contribution to the MLP output differs between the two filler conditions, replace the k highest-ranked neurons (top-k) for $k \in \{1, 4, 16, 64\}$, and evaluate on the held-out examples. The difference is highly concentrated: the highest-ranked neuron alone accounts for 41–76% of the total.

A few neurons restore accuracy to the level reached by replacing the whole MLP output (Figure 7a–c): one neuron suffices in Llama-3.1-8B (12% $\rightarrow$ 94%) and Qwen2.5-7B (48% $\rightarrow$ 98%), and four

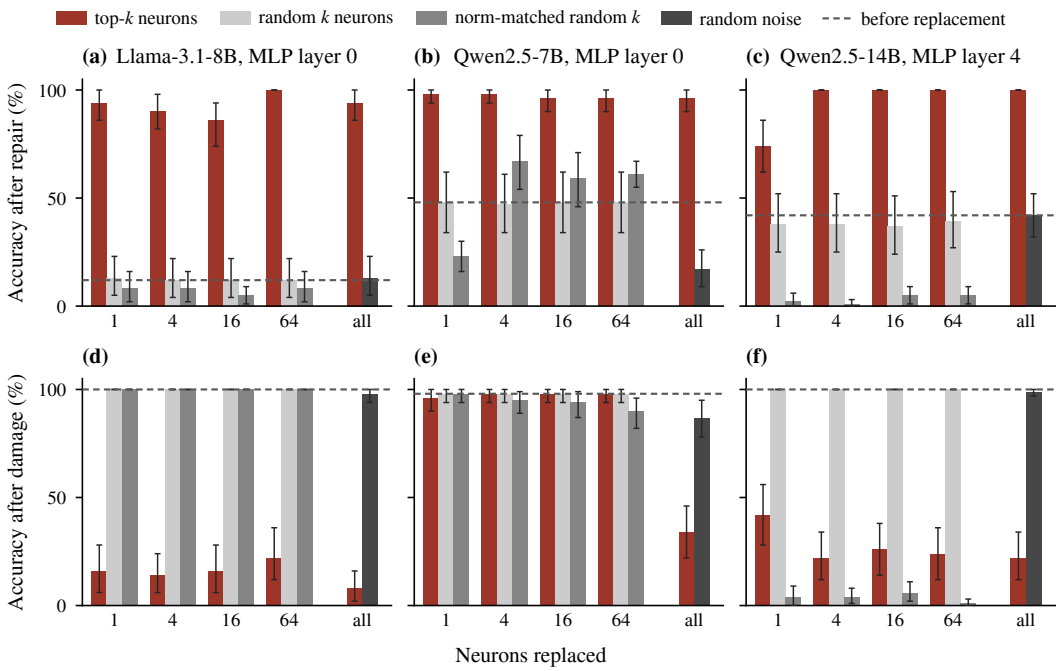

Figure 7: **Neuron-level interventions in the fixed MLP layers.** Accuracy after replacing k neurons at all filler positions, for (a–c) repair and (d–f) damage on the held-out examples ($K = 2000$). Bars compare the top-k neurons with k random neurons and k norm-matched random neurons. At $k =$ all, every neuron is replaced, which equals replacing the whole MLP output; the adjacent bar adds random noise of the same size instead. The dashed line marks the accuracy before replacement. Error bars show 95% bootstrap intervals over examples (2,000 resamples).

neurons in Qwen2.5-14B (42% $\rightarrow$ 100%). The reverse replacement with the same neurons transfers errors in Llama-3.1-8B (100% $\rightarrow$ 16%) and Qwen2.5-14B (100% $\rightarrow$ 22%), close to replacing the whole MLP output, but not in Qwen2.5-7B, where transferring errors requires more than the 64 highest-ranked neurons (Figure 7d–f).

To test whether these particular neurons matter, we repeat the replacement with randomly chosen neurons, including a version in which their change is enlarged to match that of the selected neurons ("norm-matched" in Figure 7), and we also add random noise of the same size to the MLP output. None of these alternatives restores accuracy (Figure 7a–c), and they rarely cause errors in the reverse direction. The one exception is Qwen2.5-14B, where the enlarged random changes drive accuracy close to zero in both conditions: this model is disrupted by any large change, but only the selected neurons restore correct predictions.

6 Discussion

The harm comes from a very small part of the model. Whether a filler disrupts belief tracking depends on one early MLP at the filler positions, and within it on one to four neurons (Sections 5.1 and 5.2). This changes how we should think about failures on long inputs. If length alone caused the harm (Du et al., 2025), any filler of the same length should cause similar harm. Instead, only certain repeated tokens cause harm, and the harm traces back to a few neurons that respond to these tokens and pull attention toward them. Such failures may therefore be fixed by targeting a few neurons, without changing how the model handles long inputs in general.

The failure also appears suddenly. In Llama-3.1-8B with dot filler, accuracy drops from 92% to 16% as the filler grows from 1,400 to 1,600 tokens (Figure 2). Our replacements point the same way: some of them fix the errors, yet applying them in the opposite direction does not create errors (the attention output of one early layer in Llama-3.1-8B and Qwen2.5-14B, and the selected neurons

in Qwen2.5-7B). A simple explanation is that a harmful filler builds up small changes in the early layers, and the model fails only once enough of them have built up. Removing most of these changes fixes the errors, but adding the same amount to a harmless filler is not enough to cause them. This also explains why only long runs of a harmful token lead to errors.

Which tokens are harmful also differs between models, even within one family. Qwen2.5-7B and Qwen2.5-14B use the same tokenizer and training process and differ mainly in size (Yang et al., 2024), yet white space filler harms only the 7B model and dot filler only the 14B model (Table 1). In each model, the highest-ranked neuron is active under the harmful filler and almost silent under the harmless one (Appendix H). The sensitivity therefore likely comes from what each model learned during training, not from the tokens themselves.

7 Limitations and Future Work

Our findings come from a deliberately controlled setting and explain only part of the mechanism. We test three instruction-tuned models (7–14B parameters, two families) on a single story template, insert up to 2,000 filler tokens at one position, and read answers from the logits of a fixed candidate set rather than from free-form generation. Our interventions show where filler-induced errors can be undone or induced, but not why the selected neurons respond to one filler token and not another, or how their activity draws attention to the filler.

Several experiments could address these questions. Gradually scaling the activity of the highest-ranked neuron would test whether errors appear only once the changes caused by the filler pass a certain point (Section 6). Comparing base and fine-tuned checkpoints, as Prakash et al. (2024) did for entity tracking, and tracking the selected neurons over pretraining, as Gu et al. (2025) did for tokens that draw much of the attention, could show when models become sensitive to specific tokens. Testing whether these neurons are the components that make the first token draw much of the attention (Yona et al., 2025) would show how their activity pulls attention toward the filler. Finally, adding non-lexical filler to long-context benchmarks such as RULER (Hsieh et al., 2024) and NoLiMa (Modarressi et al., 2025), which fill the context with natural text, would show whether filler-induced failures extend to other tasks, including those that require more than retrieval (Goldman et al., 2024).

8 Conclusion

Language models can be disrupted more by a long run of spaces or dots than by unrelated text of the same length, the reverse of what one would expect from a human reader. Using attention patching and activation patching, we trace this failure in belief tracking across three language models. Harmful fillers draw much of the attention of the positions that follow them, and giving a harmless prompt the attention weights of a harmful one can be enough to cause errors. This attention traces back to a single early MLP at the filler positions: exchanging its output between harmful and harmless fillers removes or induces the errors, and within it, one to four neurons suffice to restore accuracy in all three models and to induce errors in two of them. Which tokens are harmful differs across models, suggesting that this sensitivity is learned during training. Because the harm arises from a small part of the model while the task information stays fixed, these neurons may offer concrete targets for making language models robust to meaningless input.

AI use statement

We used AI tools to assist with sentence translation and searches for relevant literature. The authors reviewed and revised the AI-assisted text for accuracy and clarity. All references were manually verified by the authors. The authors take full responsibility for the final manuscript and its scientific claims.

Ethics statement

This study investigates how repeated filler tokens affect language models in controlled belief-tracking tasks. Our experiments use publicly available datasets and open-source models. The study

involves no human participants or collection of personal data. We identify no ethical concerns specific to this study.

Reproducibility statement

We use the publicly available CausalToM dataset. All model checkpoints are available on Hugging Face, including meta-llama/Llama-3.1-8B-Instruct, Qwen/Qwen2.5-7B-Instruct, and Qwen/Qwen2.5-14B-Instruct. We implement model inference and intervention experiments using PyTorch and Hugging Face Transformers. We run our experiments on NVIDIA H200, H100, and RTX PRO 6000 GPUs.

References

Matteo Bortoletto, Constantin Ruhdorfer, Lei Shi, and Andreas Bulling. Brittle minds, fixable activations: Understanding belief representations in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 22521–22543. Association for Computational Linguistics, November 2025. doi: 10.18653/v1/2025.findings-emnlp.1226. URL https://aclanthology.org/2025.findings-emnlp.1226/.

Qin Dai, Benjamin Heinzerling, and Kentaro Inui. Representational analysis of binding in language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 17468–17493. Association for Computational Linguistics, November 2024. doi: 10.18653/v1/2024.emnlp-main.967. URL https://aclanthology.org/2024.emnlp-main.967/.

Yufeng Du, Minyang Tian, Srikanth Ronanki, Subendhu Rongali, Sravan Babu Bodapati, Aram Galstyan, Azton Wells, Roy Schwartz, Eliu A Huerta, and Hao Peng. Context length alone hurts LLM performance despite perfect retrieval. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 23281–23298. Association for Computational Linguistics, November 2025. doi: 10.18653/v1/2025.findings-emnlp.1264. URL https://aclanthology.org/2025.findings-emnlp.1264/.

Jiahai Feng and Jacob Steinhardt. How do Language Models Bind Entities in Context? In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/9d1b7fc578c0d2d6431fc26d736ecaf3-Abstract-Conference.html.

Atticus Geiger, Zhengxuan Wu, Christopher Potts, Thomas Icard, and Noah Goodman. Finding Alignments Between Interpretable Causal Variables and Distributed Neural Representations. In *Proceedings of the Third Conference on Causal Learning and Reasoning*, volume 236 of *Proceedings of Machine Learning Research*, pp. 160–187. PMLR, 2024. URL https://proceedings.mlr.press/v236/geiger24a.html.

Omer Goldman, Alon Jacovi, Aviv Slobodkin, Aviya Maimon, Ido Dagan, and Reut Tsarfaty. Is it really long context if all you need is retrieval? towards genuinely difficult long context NLP. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 16576–16586. Association for Computational Linguistics, November 2024. doi: 10.18653/v1/2024.emnlp-main.924. URL https://aclanthology.org/2024.emnlp-main.924/.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, et al. The Llama 3 Herd of Models. *arXiv preprint arXiv:2407.21783*, 2024.

Xiangming Gu, Tianyu Pang, Chao Du, Qian Liu, Fengzhuo Zhang, Cunxiao Du, Ye Wang, and Min Lin. When Attention Sink Emerges in Language Models: An Empirical View. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/f1b04face60081b689ba740d39ea8f37-Abstract-Conference.html.

Yoav Gur-Arieh, Mor Geva, and Atticus Geiger. Mixing Mechanisms: How Language Models Retrieve Bound Entities In-Context. In *International Conference on Learning Representations*,

2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/2eeff35664016c7f0f8aa704f0d9a83e-Abstract-Conference.html.

Stefan Heimersheim and Neel Nanda. How to use and interpret activation patching. *arXiv preprint arXiv:2404.15255*, 2024. URL https://arxiv.org/abs/2404.15255.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, and Boris Ginsburg. RULER: What's the Real Context Size of Your Long-Context Language Models? In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=kIoBbc76Sy.

Yuping Lin, Zitao Li, Yue Xing, Pengfei He, Yingqian Cui, Yaliang Li, Bolin Ding, Jingren Zhou, and Jiliang Tang. Retrieval heads are dynamic. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15710–15729. Association for Computational Linguistics, July 2026. doi: 10.18653/v1/2026.acl-long.715. URL https://aclanthology.org/2026.acl-long.715/.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024. doi: 10.1162/tacl_a_00638. URL https://aclanthology.org/2024.tacl-1.9/.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and Editing Factual Associations in GPT. In *Advances in Neural Information Processing Systems*, volume 35, pp. 17359–17372, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/6f1d43d5a82a37e89b0665b33bf3a182-Abstract-Conference.html.

Ali Modarressi, Hanieh Deilamsalehy, Franck Dernoncourt, Trung Bui, Ryan A. Rossi, Seunghyun Yoon, and Hinrich Schuetze. NoLiMa: Long-Context Evaluation Beyond Literal Matching. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 44554–44570. PMLR, 2025. URL https://proceedings.mlr.press/v267/modarressi25a.html.

Nikhil Prakash, Tamar Rott Shaham, Tal Haklay, Yonatan Belinkov, and David Bau. Fine-Tuning Enhances Existing Mechanisms: A Case Study on Entity Tracking. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/2082273791021571c410f41d565d0b45-Abstract-Conference.html.

Nikhil Prakash, Natalie Shapira, Arnab Sen Sharma, Christoph Riedl, Yonatan Belinkov, Tamar Rott Shaham, David Bau, and Atticus Geiger. Language Models use Lookbacks to Track Beliefs. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/0d1941f73833eb107939f62d323d6cc0-Abstract-Conference.html.

Freda Shi, Xinyun Chen, Kanishka Misra, Nathan Scales, David Dohan, Ed H. Chi, Nathanael Schärli, and Denny Zhou. Large Language Models Can Be Easily Distracted by Irrelevant Context. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 31210–31227. PMLR, 2023. URL https://proceedings.mlr.press/v202/shi23a.html.

Wenhao Wu, Yizhong Wang, Guangxuan Xiao, Hao Peng, and Yao Fu. Retrieval Head Mechanistically Explains Long-Context Factuality. In *International Conference on Learning Representations*, 2025a. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/9b77f07301b1ef1fe810aae96c12cb7b-Abstract-Conference.html.

Yuheng Wu, Wentao Guo, Zirui Liu, Heng Ji, Zhaozhuo Xu, and Denghui Zhang. How large language models encode theory-of-mind: a study on sparse parameter patterns. *npj Artificial Intelligence*, 1(1):20, 2025b. doi: 10.1038/s44387-025-00031-9. URL https://www.nature.com/articles/s44387-025-00031-9.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient Streaming Language Models with Attention Sinks. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/5e5fd18f863cbe6d8ae392a93fd271c9-Abstract-Conference.html.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 Technical Report. *arXiv preprint arXiv:2412.15115*, 2024. URL https://arxiv.org/abs/2412.15115.

Itay Yona, Ilia Shumailov, Jamie Hayes, and Yossi Gandelsman. Interpreting the Repeated Token Phenomenon in Large Language Models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 72535–72555. PMLR, 2025. URL https://proceedings.mlr.press/v267/yona25a.html.

Zhongzhi Yu, Zheng Wang, Yonggan Fu, Huihong Shi, Khalid Shaikh, and Yingyan Celine Lin. Unveiling and Harnessing Hidden Attention Sinks: Enhancing Large Language Models without Training through Attention Calibration. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 57659–57677. PMLR, 2024.

A PROMPT SETTINGS

We construct stories from the CausalToM template of Prakash et al. (2026). For each story, we sample two distinct character names, two distinct containers, and two distinct contents from vocabularies of 103, 21, and 23 entries, respectively, using random seed 10, and insert them into the fixed restaurant story of Figure 8. The question asks what one character believes their own container contains; the correct answer is the content that character placed in it. Every prompt uses the same instruction. The evaluation set contains 100 prompts that all three models answer correctly without filler, 50 for each character–container pair.

Belief tracking

```
Instruction: 1. Track the belief of each character as described in
the story. 2. A character's belief is formed only when they perform
an action themselves or can observe the action taking place. 3. A
character does not have any beliefs about the container and its
contents which they cannot observe. 4. To answer the question,
predict only what is inside the queried container, strictly based on
the belief of the character, mentioned in the question. 5. If the
queried character has no belief about the container in question, then
predict 'unknown'. 6. Do not predict container or character as the
final output.

Story: Lily and Amy are working in a busy restaurant. To complete an
order, Lily grabs an opaque mug and fills it with port. Then Amy
grabs another opaque pitcher and fills it with coffee.
Question: What does Lily believe the mug contains?
Answer:
```

Figure 8: **A complete belief-tracking prompt without filler.** Lily fills the mug with port, so the correct answer is `port`. The input ends at `Answer:`.

We insert K filler tokens after the newline that ends the story and add a newline before `Question:`. We concatenate token IDs directly, so the instruction, story, and question tokens are identical across conditions. We pass prompts as plain text without a chat template; Llama prepends its default beginning-of-text token, and Qwen adds none.

We use $K \in \{0, 500, 1000, 1200, 1400, 1600, 1800, 2000\}$. Table 2 defines the seven filler types. For each model, filler type, and length, all 100 prompts receive the same filler sequence; word and letter permutations use random seed 0.

Table 2: **Filler construction.** The first four fillers repeat a single token ID K times, excluding the separating newline. The last three use Chapters I–III of *Alice's Adventures in Wonderland*, with words from the task's character, container, and state vocabularies removed; we tokenize the text with each model's tokenizer and keep its first K tokens.

Filler type	Construction
White space	Repeat the token for one space.
Dot	Repeat the period token, `.`.
Slash	Repeat the slash token, `/`.
`' hi'`	Repeat the token for `hi` preceded by a space.
Irrelevant text	Use the filtered passage in its original word order.
Shuffled text	Randomly permute the words of the filtered passage.
Anagram	Randomly permute letters within words, preserving word order, word lengths, and surrounding punctuation.

Table 3 lists the candidate answers for the evaluation in Section 2.3.

Table 3: **Candidate answers.** The set $\mathcal{C}$ is shared across models and filler conditions.

24 candidate answers					
water	milk	tea	beer	soda	juice
coffee	wine	gin	rum	champagne	cocktail
punch	espresso	cocoa	sprite	monster	bourbon
port	float	stout	ale	porter	unknown

B Additional Behavioral Results

We test whether filler-induced errors also occur when the model retrieves an explicitly stated fact. We keep the 100 stories, filler types and lengths, insertion position, candidate answers, and evaluation method of the belief-tracking experiment, and replace the instruction and question with a request for the queried container's actual contents (Figure 9). The queried character filled the queried container in every story, so the factual and belief questions share the same correct answer.

Factual question

```
Instruction: 1. Read the story carefully. 2. To answer the question,
predict only what is actually inside the queried container, strictly
based on the story. 3. Do not predict container or character as the
final output.

Story: Lily and Amy are working in a busy restaurant. To complete an
order, Lily grabs an opaque mug and fills it with port. Then Amy
grabs another opaque pitcher and fills it with coffee.
Question: What does the mug contain?
Answer:
```

Figure 9: **A complete factual-question prompt without filler.** The story matches Figure 8; the instruction and question ask for the mug's actual contents, so the correct answer remains `port`.

With 2,000 filler tokens, white space and dot filler reduce the accuracy of Llama-3.1-8B from 100% to 13% and 7%, white space filler reduces that of Qwen2.5-7B from 100% to 60%, and dot filler reduces that of Qwen2.5-14B from 97% to 35% (Figure 10; Table 4). The three text fillers retain 92–100% accuracy in all models. The disruption therefore extends to factual questions that do not require reporting a character's belief.

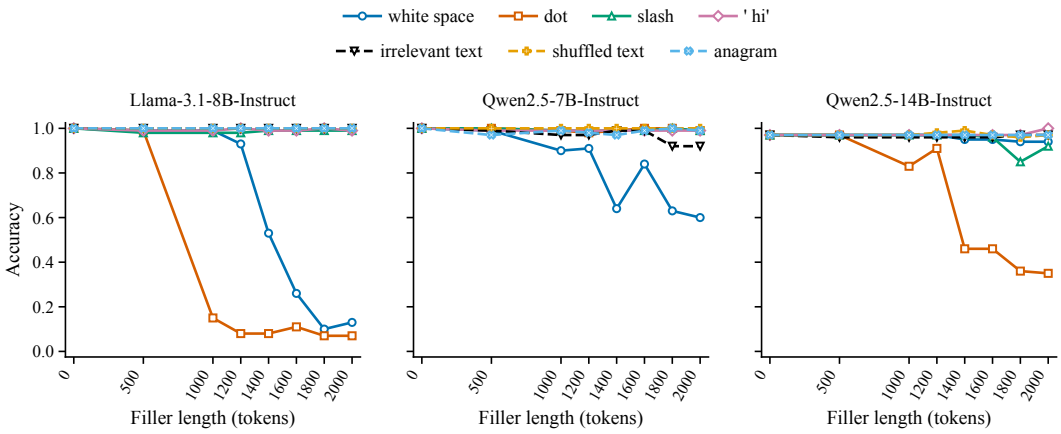

Figure 10: **Accuracy on factual questions under different filler types and lengths.** Each panel shows one model; line styles follow Figure 2.

Table 4: **Factual-question accuracy (%) without filler and with 2,000 filler tokens.**

Filler	Llama-3.1-8B	Qwen2.5-7B	Qwen2.5-14B
No filler	100	100	97
White space	13	60	94
Dot	7	99	35
Slash	99	99	92
`' hi'`	99	99	100
Irrelevant text	100	92	97
Shuffled text	100	100	97
Anagram	100	99	97

C Additional Attention Analyses

This appendix reports the attention analyses of Section 3 for all three models. Values average over attention heads and the 100 examples.

Table 5 summarizes attention from the readout position and the question tokens at $K = 2000$, and Figures 11–13 decompose the attention of the readout position and of the queried character and container tokens in each model. In every model, the degraded filler receives the largest share of attention at all three positions (72.7–92.1% in Llama-3.1-8B and Qwen2.5-14B; 31.5–35.9% in Qwen2.5-7B), whereas the other fillers leave the allocation close to that without filler.

Table 5: **Attention from the readout position and the question tokens with 2,000 filler tokens.** Percentages averaged over layers. *Filler*: total attention to filler positions, which make up 92% of the positions before the readout position. *First token*: attention to the first input token. *Answer token*: attention to the story token naming the correct answer, relative to its value without filler. Question-token values average over those tokens. Accuracy is from Table 1; bold marks degraded conditions.

			Readout attention			Question tokens
Model	Filler	Accuracy	Filler	First token	Answer token	Filler
Llama-3.1-8B	No filler	100	–	52.2	100	–
	White space	**10**	**74.6**	**1.4**	**11**	**82.4**
	Dot	**11**	**72.7**	**0.6**	**1**	**85.4**
	Slash	89	8.8	52.0	73	9.1
	`' hi'`	99	4.4	52.8	84	4.1
Qwen2.5-7B	No filler	100	–	30.2	100	–
	White space	**46**	**33.4**	**22.9**	**47**	**36.2**
	Dot	77	13.5	32.7	78	15.0
	Slash	82	10.0	30.9	82	11.4
	`' hi'`	96	6.7	32.8	85	8.8
Qwen2.5-14B	No filler	100	–	46.1	100	–
	White space	100	14.2	44.6	84	15.8
	Dot	**43**	**77.2**	**0.3**	**7**	**82.4**
	Slash	100	11.4	45.3	80	13.0
	`' hi'`	100	8.3	46.9	80	10.7

Distribution within the filler. We index the filler positions in input order as $f_0, \ldots, f_{K-1}$ and define $p_r^\ell(j) = a_{r,f_j}^\ell / A_r^\ell$, the share of filler attention that position f_j receives; $K\,p_r^\ell(j) = 1$ corresponds to a uniform distribution.

Figure 14 shows $K\,p_r^\ell(j)$ for the four degraded conditions at $K = 2000$, and Figures 15–17 show the same distributions as heat maps. In Llama-3.1-8B, dot and white space filler show the same transition at $j/K \approx 0.75$ at both reading positions. In Qwen2.5-14B, attention concentrates on alternating bands of positions beyond $j/K \approx 0.6$; in Qwen2.5-7B, it rises gradually between $j/K \approx 0.5$ and $j/K \approx 0.7$. The latter half of the filler receives 96.5% and 97.0% of the readout position's

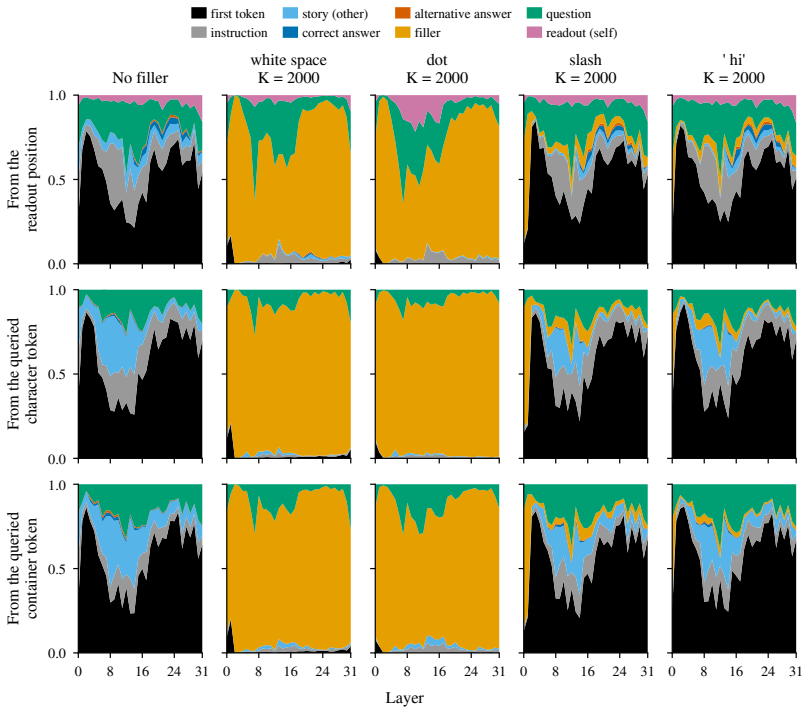

Figure 11: **Composition of attention in Llama-3.1-8B-Instruct.** Rows show attention from the readout position, the queried character token, and the queried container token; columns and colors follow Figure 3.

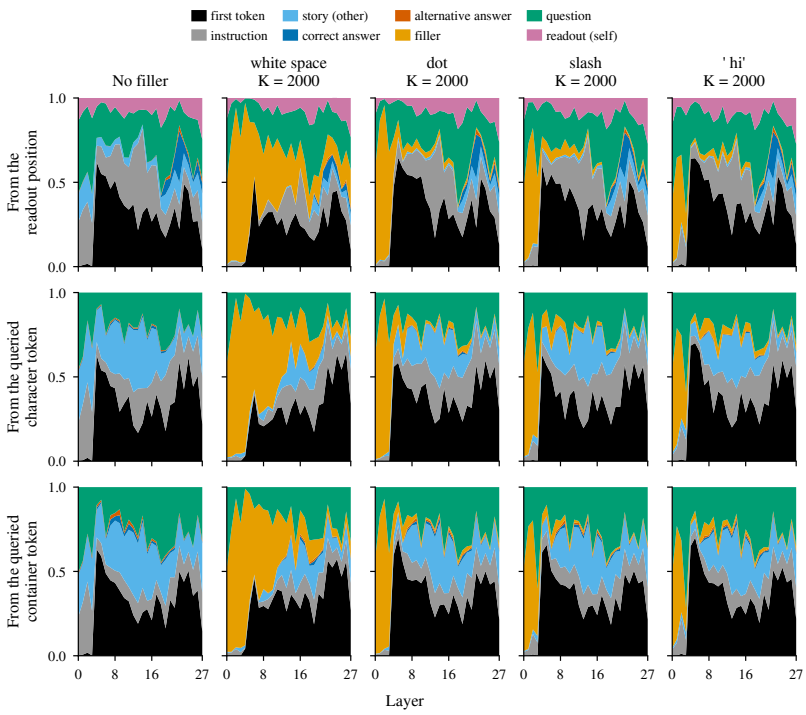

Figure 12: **Composition of attention in Qwen2.5-7B-Instruct.** Layout follows Figure 11.

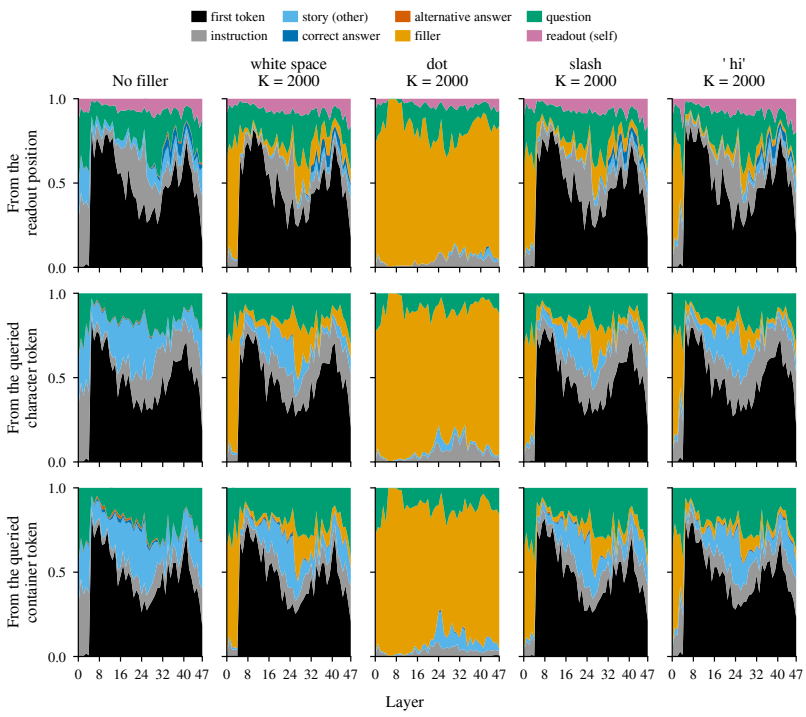

Figure 13: **Composition of attention in Qwen2.5-14B-Instruct.** Layout follows Figure 11.

attention to the filler for dot and white space filler in Llama-3.1-8B, 93.3% in Qwen2.5-14B, and 89.2% in Qwen2.5-7B.

Filler length. Figures 18–21 compare the four degraded conditions at $K \in \{500, 1000, 2000\}$. With 1,000 or fewer filler tokens, the filler receives 4.1–14.9% of readout attention, the first filler position receives far more than the uniform share, and the latter half of the filler receives 26.8–56.1% of the readout position's attention to the filler.

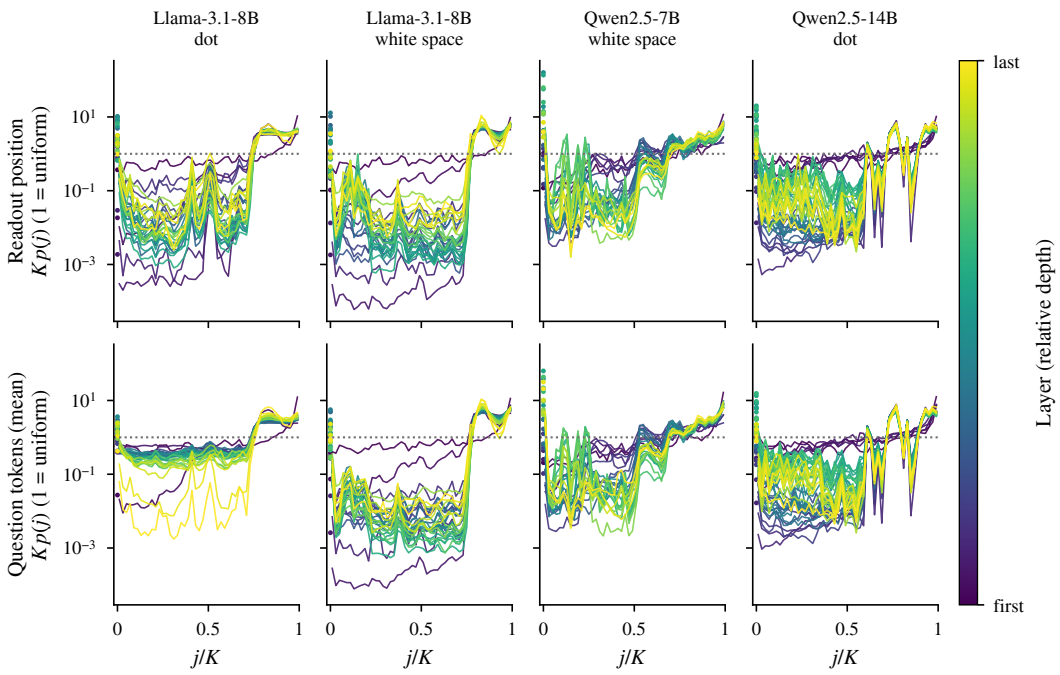

Figure 14: **Within-filler attention profiles of the four degraded conditions at $K = 2000$.** Columns show model–filler pairs. The top row shows the readout position and the bottom row the mean over question tokens. Each curve shows $K\,p_r^{\ell}(j)$ for one layer on a logarithmic scale, colored by relative depth; the dotted line at 1 marks a uniform distribution, and points at $j/K = 0$ show the first filler position. Positions are grouped into 50 equal-width bins.

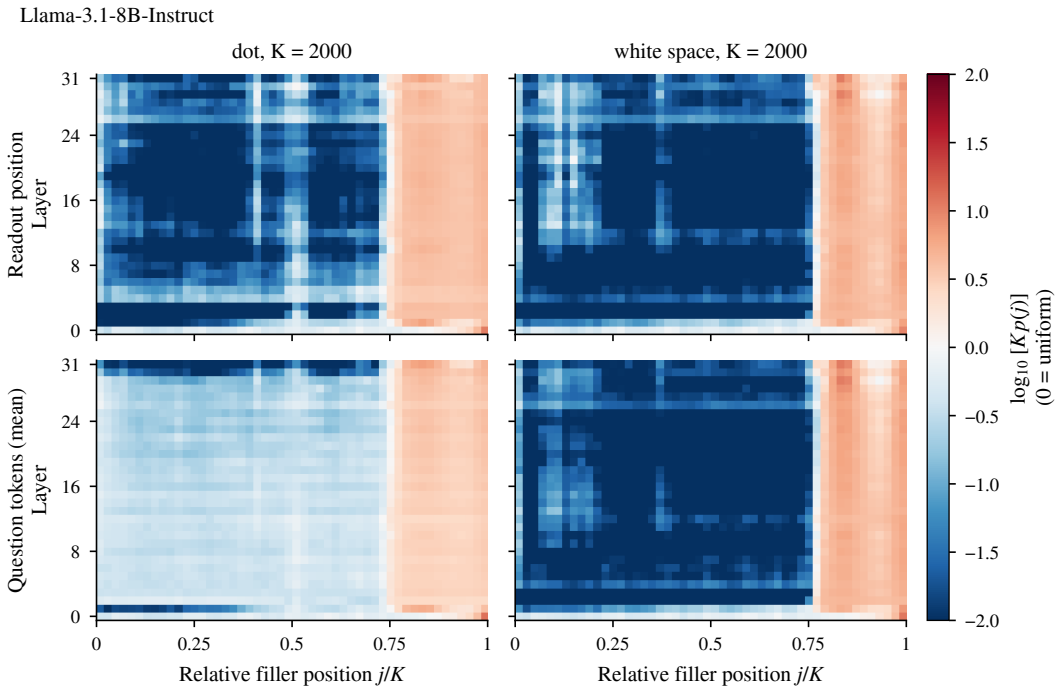

Figure 15: **Within-filler attention in Llama-3.1-8B-Instruct at $K = 2000$.** Columns show dot and white space filler; rows show the readout position and the mean over question tokens. Color encoding follows Figure 4.

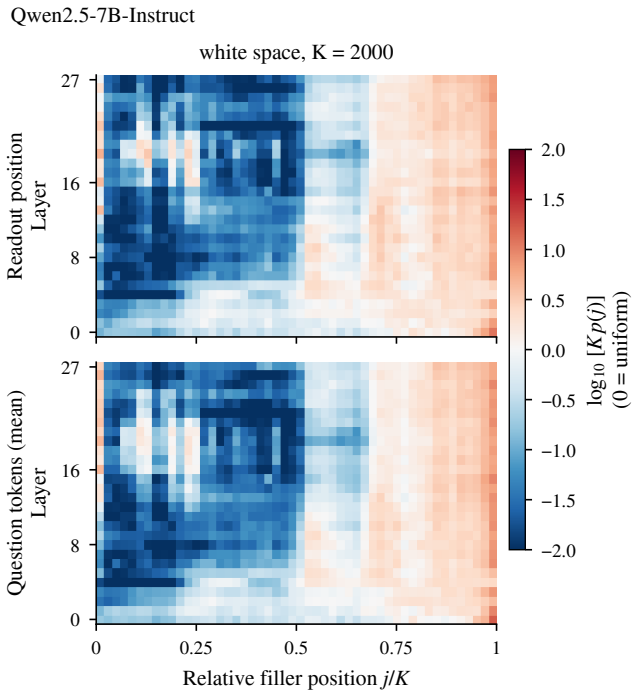

Figure 16: **Within-filler attention in Qwen2.5-7B-Instruct with white space filler at $K = 2000$.** Rows and color encoding follow Figure 15.

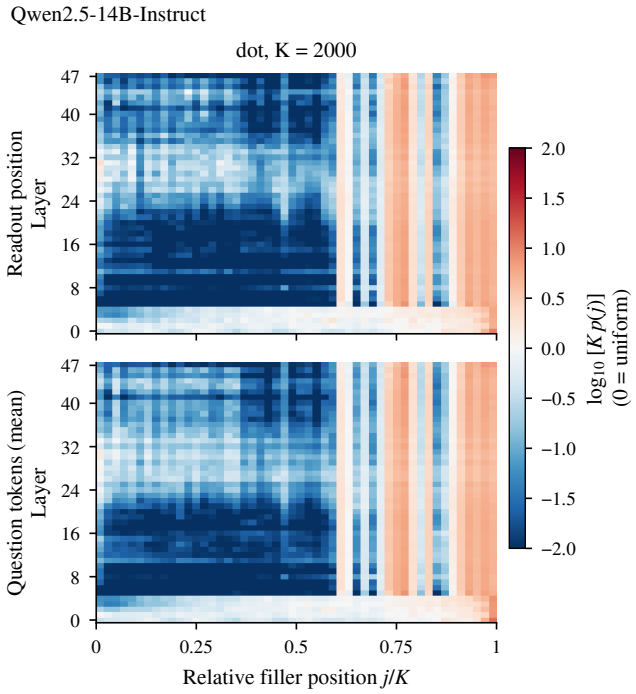

Figure 17: **Within-filler attention in Qwen2.5-14B-Instruct with dot filler at $K = 2000$.** Rows and color encoding follow Figure 15.

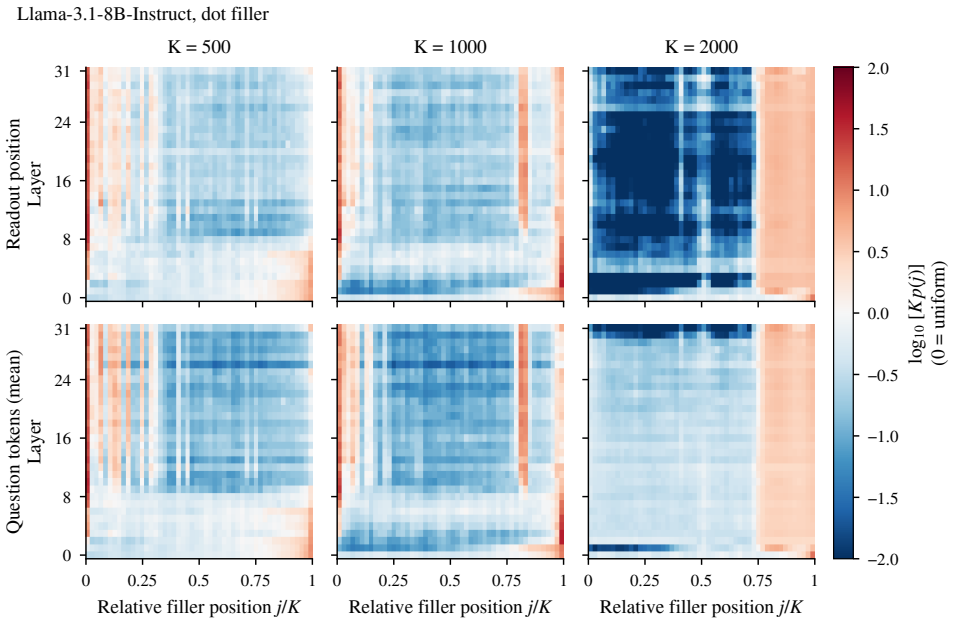

Figure 18: **Within-filler attention for dot filler in Llama-3.1-8B-Instruct at** $K \in \{500, 1000, 2000\}$**.** Rows and color encoding follow Figure 15.

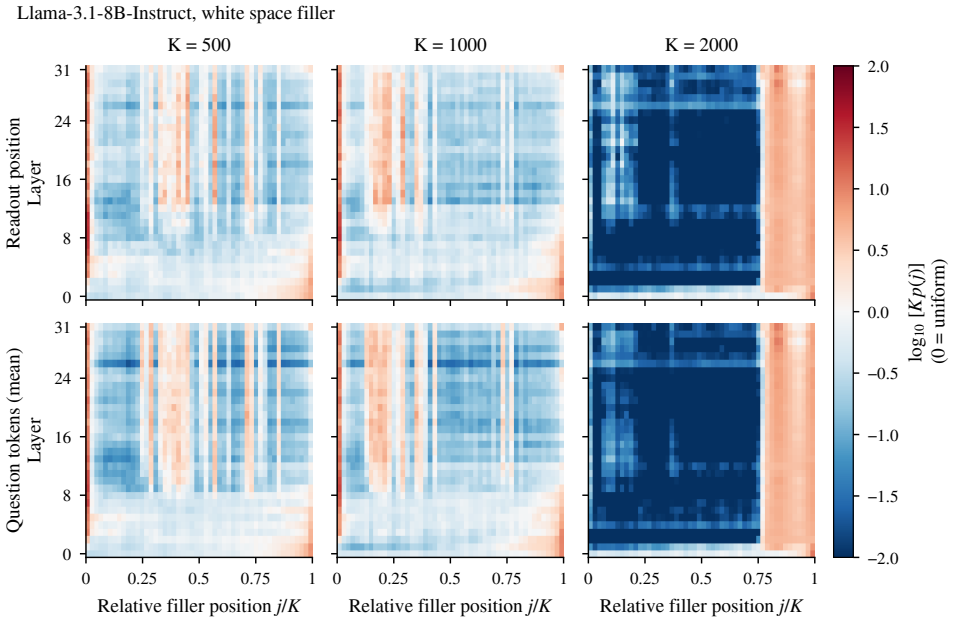

Figure 19: **Within-filler attention for white space filler in Llama-3.1-8B-Instruct at** $K \in \{500, 1000, 2000\}$**.** Rows and color encoding follow Figure 15.

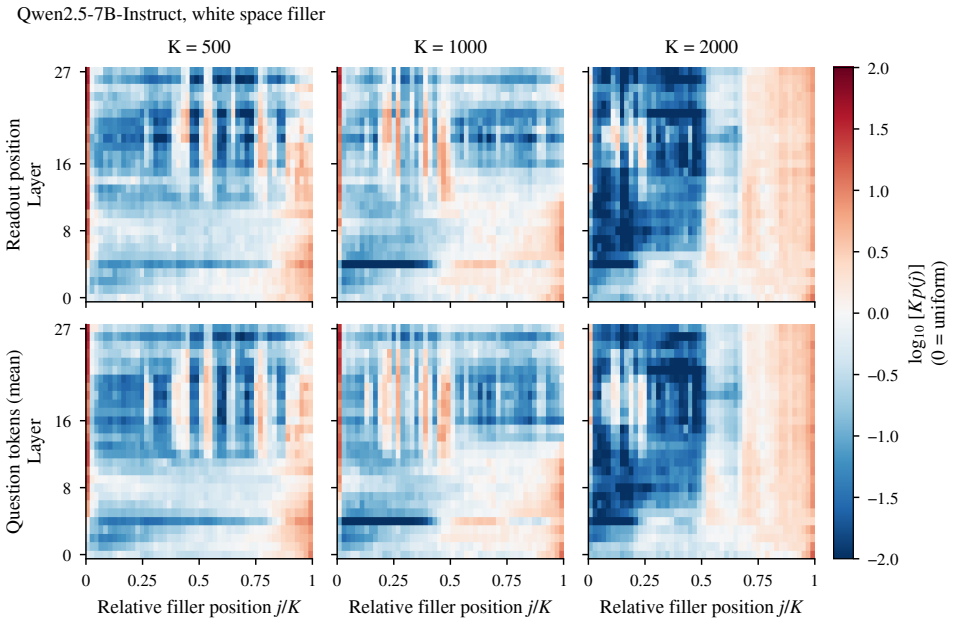

Figure 20: **Within-filler attention for white space filler in Qwen2.5-7B-Instruct at** $K \in \{500, 1000, 2000\}$**.** Rows and color encoding follow Figure 15.

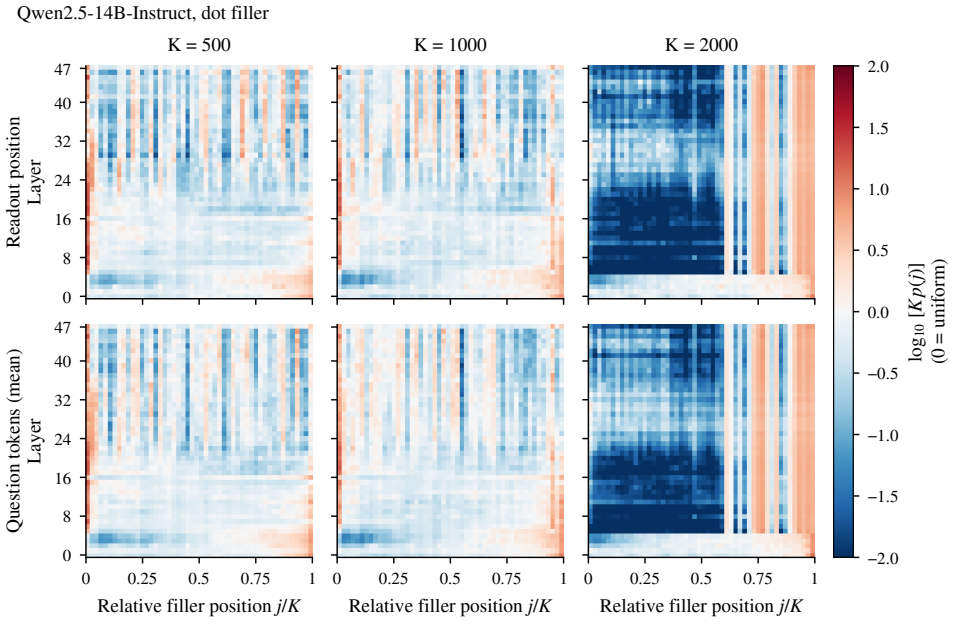

Figure 21: **Within-filler attention for dot filler in Qwen2.5-14B-Instruct at** $K \in \{500, 1000, 2000\}$**.** Rows and color encoding follow Figure 15.

D FILLER-KEY PATCHING

We test whether key vectors at filler positions can transfer the effects of filler type. Keys determine how strongly positions match attention queries. Source and target prompts share the story, question, and filler length, and differ only in filler type. Let D denote an unmodified source run and R the target run with interventions. Target quantities always refer to the current forward pass, including the effects of earlier interventions. Let $\mathcal{F}$ contain the filler positions and $\mathcal{P}$ all later positions (the separator, question, and `Answer:` prefix). At each layer ℓ, we replace keys at positions $j \in \mathcal{J}$, where $\mathcal{J} \in \{\mathcal{F}, \mathcal{P}, \mathcal{F} \cup \mathcal{P}\}$:

$$\widetilde{\mathbf{k}}_{\ell,h,j} = \begin{cases} \mathbf{k}^{D}_{\ell,h,j}, & j \in \mathcal{J}, \\ \mathbf{k}^{R}_{\ell,h,j}, & j \notin \mathcal{J}. \end{cases} \tag{D.1}$$

Here h indexes key heads and j token positions. We replace all key heads after rotary position encoding; the target run supplies its own queries and values and applies its usual causal mask. We intervene jointly at all 32, 28, or 48 layers of Llama-3.1-8B, Qwen2.5-7B, or Qwen2.5-14B, with 2,000 filler tokens and the 100 evaluation examples.

Replacing the keys of white space filler with those of `'hi'` filler raises Qwen2.5-7B accuracy from 46% to 92%, and the reverse replacement lowers it from 95% to 2% (Table 6). In Qwen2.5-14B, replacing the keys of dot filler with those of `' hi'` filler raises accuracy from 43% to 100%, and the reverse replacement lowers it from 100% to 0%. Because the target run computes its own queries and values, filler keys alone transfer the effect of filler type in both Qwen models.

Table 6: **Accuracy after replacing key vectors in different input regions.** Accuracy (%) over 100 examples with 2,000 filler tokens. Source → target: the filler conditions supplying and receiving the keys. *Unpatched*: target accuracy without intervention in this experiment. *Filler*, *Post-filler*, and *Both*: keys replaced at the filler positions, at all later positions, or at both, jointly at every layer.

Model	Source → target	Unpatched	Filler	Post-filler	Both
Repair direction: higher-accuracy source → lower-accuracy target					
Llama-3.1-8B	`' hi'` → dot	11	3	0	3
	`' hi'` → white space	10	3	12	3
Qwen2.5-7B	`' hi'` → dot	76	94	91	97
	`' hi'` → slash	85	89	92	95
	`' hi'` → white space	46	92	53	97
Qwen2.5-14B	`' hi'` → dot	43	100	57	94
	slash → dot	43	98	55	99
	white space → dot	43	91	57	88
Damage direction: lower-accuracy source → higher-accuracy target					
Llama-3.1-8B	dot → `' hi'`	99	3	0	3
	white space → `' hi'`	99	3	29	4
Qwen2.5-7B	dot → `' hi'`	95	12	85	24
	slash → `' hi'`	95	96	90	96
	white space → `' hi'`	95	2	87	3
Qwen2.5-14B	dot → `' hi'`	100	0	88	6
	dot → slash	100	5	90	6
	dot → white space	100	0	94	0

E DEPENDENCE ON PRECEDING CONTEXT

We test whether the accuracy loss requires filler representations computed from the task story. Write the original input and the source input as

$$x_R = b \,\|\, p_R \,\|\, f \,\|\, q, \qquad x_D = b \,\|\, p_D \,\|\, f, \qquad |p_D| = |p_R|, \tag{E.1}$$

where $\|$ denotes token concatenation, b is the original special-token prefix, p_R contains the instruction and story through the newline before filler, f is the repeated filler, and q contains the separator, question, and answer prefix. We construct p_D by tokenizing a geology passage and truncating it to $|p_R|$ tokens. Thus, filler token identities and absolute positions match across inputs.

Let $\mathbf{h}^D_{\ell,j}$ be the hidden state at position j after block ℓ in an unmodified run on x_D, and let $\mathbf{h}^R_{\ell,j}$ be the corresponding output in the intervened run on x_R. After every transformer block, we replace

$$\widetilde{\mathbf{h}}_{\ell,j} = \begin{cases} \mathbf{h}^D_{\ell,j}, & j \in \mathcal{F}, \\ \mathbf{h}^R_{\ell,j}, & j \notin \mathcal{F}, \end{cases} \qquad \ell = 0, \ldots, L-1, \tag{E.2}$$

where $\mathcal{F}$ is the set of filler positions and L is the number of transformer blocks. We compare this intervention with no replacement and with the same replacement using states from an unmodified run on x_R. Each condition uses the 100 evaluation examples at filler lengths of 500 and 2,000 tokens.

With states computed after unrelated text, accuracy remains 99–100% at 500 filler tokens but falls at 2,000 tokens to 7% for dot and white space filler in Llama-3.1-8B, 0% for white space filler in Qwen2.5-7B, and 10% for dot filler in Qwen2.5-14B (Table 7). Replacement with states from the same input reproduces the unmodified predictions on every example. Filler representations computed after unrelated text thus reproduce the accuracy loss of long filler; the loss does not require story-specific information in these representations.

Table 7: **Accuracy after replacing filler representations computed from the preceding context.** Accuracy (%) over 100 examples. *No replacement*: the original task prompt. *Same input*: hidden states at every filler position, after every block, replaced with those from a separate run on the identical input. *Unrelated text*: states from the same filler tokens preceded by a geology passage of equal token length.

Model	Filler	K	No replacement	Same input	Unrelated text
Llama-3.1-8B	Dot	500	100	100	100
	Dot	2,000	11	11	7
	White space	500	100	100	100
	White space	2,000	10	10	7
Qwen2.5-7B	White space	500	100	100	99
	White space	2,000	46	46	0
Qwen2.5-14B	Dot	500	100	100	100
	Dot	2,000	43	43	10

F Attention Patching

We test how attention at the readout position r, the final token of `Answer:`, transfers the effects of filler type. Prompts, examples, and notation follow Appendix D: source and target prompts differ only in filler type, D denotes an unmodified source run, and R the current intervened target run.

We compare interventions at three selected layers and at all layers. To select layers, we measure each layer's accuracy change in both directions for white space versus `' hi'` in Llama-3.1-8B and Qwen2.5-7B, and for dot versus white space in Qwen2.5-14B. Among layers whose 95% bootstrap interval (2,000 resamples of the 100 examples, seed 0) excludes zero in at least one direction, we select the three with the largest summed absolute change. Table 8 evaluates the selected layers on other filler pairs. Layers are indexed from 0.

F.1 Filler-Score Patching

Attention scores are the scaled query–key dot products before softmax; attention weights are the normalized values after softmax. For run $X \in \{D, R\}$, these quantities at readout position r are

$$s^X_{\ell,h,j} = \frac{(\mathbf{q}^X_{\ell,h,r})^\top \mathbf{k}^X_{\ell,g(h),j}}{\sqrt{d}}, \qquad a^X_{\ell,h,j} = \frac{\exp(s^X_{\ell,h,j})}{\sum_{u=0}^{r} \exp(s^X_{\ell,h,u})}. \tag{F.1}$$

Here ℓ indexes layers, h indexes query heads, $j \in \{0, \ldots, r\}$ indexes input positions, and d is the head dimension. The vectors $\mathbf{q}$ and $\mathbf{k}$ are queries and keys after rotary position encoding. The

Table 8: **Accuracy after replacing readout attention across filler conditions.** Accuracy (%) over 100 examples with 2,000 filler tokens. *Unpatched*: target accuracy without intervention in this experiment. *Filler scores* replaces the pre-softmax scores at filler positions and renormalizes the row; *Full-row weights* replaces the entire post-softmax row. Both act on every attention head at the readout position and use the target's values. *Three*: the three layers selected for each intervention; *All*: all 32, 28, or 48 layers.

			Filler scores		Full-row weights	
Model	Source → target	Unpatched	Three	All	Three	All
Repair direction: higher-accuracy source → lower-accuracy target						
Llama-3.1-8B	`' hi'` → dot	11	10	4	15	40
Qwen2.5-7B	`' hi'` → dot	76	83	90	92	95
	`' hi'` → slash	85	81	82	92	95
Qwen2.5-14B	`' hi'` → dot	43	21	18	70	100
	slash → dot	43	16	29	54	99
Damage direction: lower-accuracy source → higher-accuracy target						
Llama-3.1-8B	dot → `' hi'`	99	0	0	1	1
Qwen2.5-7B	dot → `' hi'`	95	92	98	87	76
	slash → `' hi'`	95	93	87	88	86
Qwen2.5-14B	dot → `' hi'`	100	100	8	99	9
	dot → slash	100	100	2	100	49

model's head mapping $g(h)$ associates each query head with its key/value head; multiple query heads share a key/value head in grouped-query attention.

We replace scores at filler positions $\mathcal{F}$, normalize the full row, and compute the head output using the target values $\mathbf{v}^R$:

$$\widetilde{s}_{\ell,h,j} = \begin{cases} s^D_{\ell,h,j}, & j \in \mathcal{F}, \\ s^R_{\ell,h,j}, & j \notin \mathcal{F}, \end{cases}$$
$$\widetilde{\mathbf{a}}_{\ell,h} = \mathrm{softmax}(\widetilde{\mathbf{s}}_{\ell,h}), \qquad \widetilde{\mathbf{o}}_{\ell,h,r} = \sum_{j=0}^{r} \widetilde{a}_{\ell,h,j} \mathbf{v}^R_{\ell,g(h),j}. \tag{F.2}$$

Renormalization changes the total attention assigned to non-filler positions while preserving their relative weights within that row.

The three selected layers are $\{1, 25, 27\}$, $\{4, 5, 27\}$, and $\{8, 10, 26\}$ for Llama-3.1-8B, Qwen2.5-7B, and Qwen2.5-14B. Replacing filler scores at all layers transfers errors from dot to `' hi'` filler in Llama-3.1-8B (99% to 0%) and Qwen2.5-14B (100% to 8%). The scores assigned to filler at the readout position thus transmit the disruptive effect.

F.2 Full-Row Attention-Weight Patching

We replace the complete post-softmax attention row with the source weights:

$$\widetilde{a}_{\ell,h,j} = a^D_{\ell,h,j}, \qquad \widetilde{\mathbf{o}}_{\ell,h,r} = \sum_{j=0}^{r} a^D_{\ell,h,j} \mathbf{v}^R_{\ell,g(h),j}. \tag{F.3}$$

This replacement covers every input position and uses the source distribution directly, without another softmax. All other positions follow the target computation.

The three selected layers are $\{1, 13, 15\}$, $\{4, 19, 22\}$, and $\{8, 36, 39\}$ for Llama-3.1-8B, Qwen2.5-7B, and Qwen2.5-14B. With replacements at all layers, the weights of `' hi'` filler raise accuracy on dot filler from 11% to 40% in Llama-3.1-8B, from 76% to 95% in Qwen2.5-7B, and from 43% to 100% in Qwen2.5-14B, and the reverse replacements lower accuracy to 1%, 76%, and 9%. The full attention row of the readout position therefore transfers the effect in both directions.

G MODULE-LEVEL INTERVENTIONS

This appendix details the interventions of Section 5.1. Unless stated otherwise, $K = 2000$, replacements cover all filler positions, and entries are accuracies (%) over 50 examples.

Implementation. For each example, we run the source condition and store, at every layer and filler position, the output of the attention module (after its output projection) and of the MLP (after its down projection). We then run the target condition and overwrite the selected module's output at the filler positions of one layer; positions after the filler are never replaced. Replacing a condition's outputs with its own stored outputs reproduces its logits exactly (five examples, three layers, and both module types per model). We split the 100 examples with a fixed seed into a localization half and a held-out half, each with 25 questions per character–container pair. Intervals are 95% percentile intervals from 2,000 paired bootstrap resamples of examples.

Layer selection. On the localization half, a layer qualifies if its repair or damage effect has an interval excluding zero and a magnitude of at least half the accuracy gap between the two conditions. The held-out evaluation covers the qualifying layers and their immediate neighbors for each module type. For the bidirectional test in Figure 6d–f, we fix the MLP layer whose repair and damage effects both qualify with the largest magnitude: layer 0 in Llama-3.1-8B and Qwen2.5-7B and layer 4 in Qwen2.5-14B.

Localization sweep. Table 9 lists every layer whose replacement changes accuracy with an interval excluding zero. Repair succeeds only within layers 0–4. In the damage direction, single-layer MLP outputs transfer errors in all three models, whereas single-layer attention outputs do so only in Qwen2.5-7B. Replacements at several other layers lower accuracy instead; for example, the intact attention output of layer 1 or the intact MLP output of layer 4 reduces Qwen2.5-7B accuracy to 0%.

Table 9: **Layers with a significant accuracy change in the localization sweep.** Entries list layer: accuracy after replacement (%) for every layer whose 95% paired bootstrap interval excludes zero; – indicates none. Repair replaces the outputs of the degraded condition (dot, white space, and dot in Llama-3.1-8B, Qwen2.5-7B, and Qwen2.5-14B) with those of the intact condition (`' hi'`, `' hi'`, and white space); damage applies the reverse. Localization half.

Model	Direction	Module	Unpatched	Layer: accuracy after replacement
Llama-3.1-8B	Repair	Attention	10	0: 92
	Repair	MLP	10	0: 96, 1: 38
	Damage	Attention	98	–
	Damage	MLP	98	0: 6, 1: 12
Qwen2.5-7B	Repair	Attention	44	0: 94, 1: 0, 2: 80, 3: 96, 4: 0, 6: 36
	Repair	MLP	44	0: 96, 1: 96, 2: 90, 3: 88, 4: 0, 5: 16, 7: 24, 10: 36, 11: 36
	Damage	Attention	92	0: 44, 2: 82, 8: 84
	Damage	MLP	92	0: 34, 3: 66
Qwen2.5-14B	Repair	Attention	44	0: 100, 1: 24, 2: 28, 3: 100, 4: 86, 5: 26, 6: 36, 8: 28, 21: 36
	Repair	MLP	44	0: 100, 1: 72, 2: 64, 3: 100, 4: 100, 5: 12, 6: 20, 12: 32, 15: 34, 18: 32, 25: 30
	Damage	Attention	100	–
	Damage	MLP	100	4: 32, 5: 78, 6: 64

Held-out evaluation. Table 10 reports all evaluated layers on the held-out half. Every effect that meets the selection criterion recurs with the same sign.

Filler length. Table 11 repeats the fixed-layer replacements at $K \in \{1400, 1600, 2000\}$. Repair restores accuracy at every length. In Llama-3.1-8B, the dot output transfers errors already at $K = 1400$, where the dot condition itself still reaches 94%; in Qwen2.5-14B, the transferred error grows with filler length, and in Qwen2.5-7B it appears only at $K = 2000$.

Table 10: **Held-out evaluation of all candidate layers.** Entries list layer: accuracy after replacement (%). Bold marks changes whose 95% paired bootstrap interval excludes zero. Pairs and directions follow Table 9; held-out half.

Model	Direction	Module	Unpatched	Layer: accuracy after replacement
Llama-3.1-8B	Repair	Attention	12	**0: 96**, 1: 8
	Repair	MLP	12	**0: 94**, **1: 36**, 2: 12
	Damage	Attention	100	0: 100, 1: 100
	Damage	MLP	100	**0: 8**, **1: 18**, 2: 100
Qwen2.5-7B	Repair	Attention	48	**0: 96**, **1: 0**, **2: 90**, **3: 98**, **4: 2**
	Repair	MLP	48	**0: 96**, **1: 98**, **2: 96**, **3: 96**, **4: 0**
	Damage	Attention	98	**0: 46**, 1: 98, **2: 82**, **3: 90**, 4: 92
	Damage	MLP	98	**0: 34**, **1: 88**, 2: 96, **3: 64**, 4: 92
Qwen2.5-14B	Repair	Attention	42	**0: 100**, **1: 10**, 2: 28, **3: 100**, **4: 84**, **5: 18**
	Repair	MLP	42	**0: 100**, **1: 78**, **2: 68**, **3: 100**, **4: 100**, **5: 14**, **6: 14**, 7: 50
	Damage	Attention	100	0: 100, 1: 100, 2: 100, 3: 100, 4: 100, 5: 100
	Damage	MLP	100	0: 100, 1: 100, 2: 100, 3: 100, **4: 22**, **5: 88**, **6: 78**, 7: 100

Table 11: **Fixed-layer MLP replacement across filler lengths.** Entries give accuracy before → after replacement (%). The first row of each model is repair and the second damage; held-out half.

Model	Layer	Source → target	$K = 1400$	$K = 1600$	$K = 2000$
Llama-3.1-8B	0	`' hi'` → dot	94 → 100	20 → 94	12 → 94
		dot → `' hi'`	100 → 8	100 → 10	100 → 8
Qwen2.5-7B	0	`' hi'` → white space	68 → 96	54 → 96	48 → 96
		white space → `' hi'`	96 → 88	94 → 90	98 → 34
Qwen2.5-14B	4	white space → dot	86 → 100	84 → 100	42 → 100
		dot → white space	100 → 90	100 → 70	100 → 22

Other filler pairs. Table 12 applies the fixed-layer MLP replacement to the remaining filler pairs. White space in Llama-3.1-8B behaves like dot. Fillers that remain intact at reduced accuracy, slash in Llama-3.1-8B and dot and slash in Qwen2.5-7B, gain accuracy from the `' hi'` output, but their own outputs do not transfer errors. In Qwen2.5-14B, the dot output transfers errors to both slash and `' hi'` filler, and both of their outputs repair dot filler.

Table 12: **Fixed-layer MLP replacement for other filler pairs.** Entries give accuracy before → after replacement (%); held-out half.

Model	Layer	Source → target	Accuracy
Llama-3.1-8B	0	slash → `' hi'`	100 → 100
		white space → `' hi'`	100 → 14
		`' hi'` → slash	88 → 100
		`' hi'` → white space	12 → 96
Qwen2.5-7B	0	dot → `' hi'`	98 → 96
		slash → `' hi'`	98 → 98
		`' hi'` → dot	78 → 98
		`' hi'` → slash	88 → 96
Qwen2.5-14B	4	dot → `' hi'`	100 → 6
		dot → slash	100 → 40
		`' hi'` → dot	42 → 100
		slash → dot	42 → 100

Filler positions. Table 13 restricts the fixed-layer replacement to the latter part of the filler, positions $j \geq j^*$ (tail), or to an equally long block immediately before it (front). We set j^* to the median

over examples of the first filler position at which the residual-stream norm of the degraded condition exceeds ten times its median over the first 500 filler positions for at least 20 consecutive positions, at the earliest layer where at least half of the examples show such a position; when no layer qualifies, as in Qwen2.5-7B, $j^* = 0.75K$. Repair in Llama-3.1-8B and Qwen2.5-14B depends mainly on the tail, whereas damage in Llama-3.1-8B succeeds from either block. In Qwen2.5-7B, neither block alone suffices in either direction.

Table 13: **Fixed-layer MLP replacement at subsets of filler positions.** *All* replaces every filler position; *Front* and *Tail* replace the blocks before and from filler position j^*, each of length $K - j^*$. Held-out half.

Model	j^*	Source → target	Unpatched	All	Front	Tail
Llama-3.1-8B	1479	`' hi'` → dot	12	94	14	88
		dot → `' hi'`	100	8	10	22
Qwen2.5-7B	1500	`' hi'` → white space	48	96	54	66
		white space → `' hi'`	98	34	94	82
Qwen2.5-14B	1510	white space → dot	42	100	50	94
		dot → white space	100	22	86	44

All-layer replacement. Table 14 compares the fixed single-layer MLP replacement with replacing the outputs of one module type in all layers at once. Replacing all MLP outputs is comparable to replacing the fixed layer. Replacing all attention outputs repairs both Qwen models but not Llama-3.1-8B, and it transfers errors in all three models, although no single attention layer does so in Llama-3.1-8B and Qwen2.5-14B.

Table 14: **Single-layer versus all-layer replacement.** Accuracy (%) on the held-out half.

			MLP		Attention
Model	Source → target	Unpatched	Fixed layer	All layers	All layers
Llama-3.1-8B	`' hi'` → dot	12	94	98	8
	dot → `' hi'`	100	8	16	44
Qwen2.5-7B	`' hi'` → white space	48	96	96	98
	white space → `' hi'`	98	34	46	0
Qwen2.5-14B	white space → dot	42	100	100	100
	dot → white space	100	22	12	2

Internal read-outs. Table 15 reports the readout position's attention to the filler and the norm ratio of the final-layer residual stream for the fixed-layer replacements. Both move with accuracy: repair lowers the readout position's attention to the filler to at most 8%, and damage raises it to 44–92%.

Table 15: **Internal read-outs of the fixed-layer MLP replacements.** Entries give values before → after replacement. *Readout attention* is the readout position's attention to the filler (%), averaged over attention heads, the three layers selected in Appendix F.1, and examples. *Norm ratio* is the largest final-layer residual-stream norm at the filler positions divided by its median over the first 500 filler positions, averaged over examples. Held-out half.

Model	Source → target	Accuracy	Readout attention	Norm ratio
Llama-3.1-8B	`' hi'` → dot	12 → 94	92 → 3	4.7 → 1.2
	dot → `' hi'`	100 → 8	3 → 92	1.4 → 4.1
Qwen2.5-7B	`' hi'` → white space	48 → 96	59 → 1	2.8 → 1.2
	white space → `' hi'`	98 → 34	1 → 44	1.4 → 1.9
Qwen2.5-14B	white space → dot	42 → 100	86 → 8	3.9 → 1.5
	dot → white space	100 → 22	8 → 88	1.4 → 4.6

H NEURON-LEVEL INTERVENTIONS

This appendix details the interventions of Section 5.2. The setup follows Appendix G: filler length $K = 2000$, the fixed MLP layers, replacements at all filler positions, the same localization and held-out halves, and the same bootstrap intervals. Entries are accuracies (%) on the held-out half ($n = 50$).

Neurons and replacement. Let $\mathbf{m}_p \in \mathbb{R}^{d_{\text{ff}}}$ be the input to the output projection W_{out} of the MLP at filler position p, so that the MLP output is $W_{\text{out}}\mathbf{m}_p = \sum_i m_{p,i}\mathbf{w}_i$, where $\mathbf{w}_i$ is the i-th column of W_{out}. For a set of neurons $\mathcal{S}$, we store $\mathbf{m}_p$ in the source condition and, in the target condition, overwrite the components in $\mathcal{S}$ at every filler position before the output projection. Replacing all components is equivalent to replacing the whole MLP output (Appendix G). At position p, the replacement changes the MLP output by $\sum_{i\in\mathcal{S}}(m^{\text{source}}_{p,i} - m^{\text{target}}_{p,i})\mathbf{w}_i$; we measure the size of this change over all filler positions by

$$\Delta(\mathcal{S}) = \Big(\sum_{p\in\mathcal{F}}\Big\|\sum_{i\in\mathcal{S}}\big(m^{\text{source}}_{p,i} - m^{\text{target}}_{p,i}\big)\mathbf{w}_i\Big\|^2\Big)^{1/2}.$$

Ranking. We rank neurons once, on the localization half with $K = 2000$. The score of neuron i is the squared difference between the two filler conditions in its contribution to the MLP output, summed over filler positions and averaged over examples:

$$\rho_i = \frac{1}{|\mathcal{D}_{\text{loc}}|}\sum_{x\in\mathcal{D}_{\text{loc}}}\sum_{p\in\mathcal{F}}\big(m^{\text{intact}}_{p,i}(x) - m^{\text{degraded}}_{p,i}(x)\big)^2\,\|\mathbf{w}_i\|^2.$$

The top-k set $\mathcal{S}_k$ contains the k neurons with the largest scores and is used in both directions. The ranking is concentrated (Table 16): the highest-ranked neuron accounts for 41–76% of the total score and, replaced alone, produces 53–64% of the change in the MLP output caused by replacing all neurons. In Qwen2.5-14B, the two highest-ranked neurons account for 99.9% of the score. The four highest-ranked neurons are 8268, 5971, 5590, and 6966 in Llama-3.1-8B; 9335, 11030, 3123, and 13465 in Qwen2.5-7B; and 11696, 1368, 13118, and 418 in Qwen2.5-14B.

Table 16: **Concentration of the neuron ranking.** *Score share* is the fraction of the total score $\sum_i \rho_i$ held by the top-k neurons. *Output change* is the change in the MLP output from replacing the top-k neurons, $\Delta(\mathcal{S}_k)$, as a percentage of the change from replacing all neurons, averaged over the held-out half. Entries are percentages.

		Score share, $k =$				Output change, $k =$			
Model	d_{ff}	1	4	16	64	1	4	16	64
Llama-3.1-8B	14,336	41	51	58	68	53	59	69	76
Qwen2.5-7B	18,944	46	58	68	78	57	65	73	83
Qwen2.5-14B	13,824	76	100	100	100	64	100	100	100

Activity of the selected neurons. Table 17 reports the activation of the highest-ranked neuron. In each model, this neuron is active under the degraded filler and nearly silent under the intact filler, and its activation grows with filler length. Some lower-ranked neurons show the opposite pattern: in Qwen2.5-7B, the neurons ranked second to fourth are active only under the intact filler. In Qwen2.5-14B, the second-ranked neuron behaves like the first and reaches a mean activation of 137 under dot filler at $K = 2000$.

Controls. *Random k* replaces k randomly drawn neurons, with two fixed draws per k used for all examples. *Norm-matched random k* uses the same neurons $\mathcal{R}$ but scales up their change,

$$m_{p,i} \leftarrow m^{\text{target}}_{p,i} + \alpha\big(m^{\text{source}}_{p,i} - m^{\text{target}}_{p,i}\big), \qquad i \in \mathcal{R},\ p \in \mathcal{F}, \qquad \alpha = \frac{\Delta(\mathcal{S}_k)}{\Delta(\mathcal{R})},$$

so that the MLP output changes as much as under the top-k replacement; α is computed per example. Because random neurons differ little between the two conditions, α is large, with a median of 77,

Table 17: **Activation of the highest-ranked neuron.** Mean absolute activation $|m_{p,i}|$ over the last quarter of the filler positions under the degraded and the intact filler, averaged over the localization half.

Model	Neuron	Filler	$K = 1400$	$K = 1600$	$K = 2000$
Llama-3.1-8B	8268	dot	0.59	0.80	1.29
		' hi'	0.01	0.01	0.01
Qwen2.5-7B	9335	white space	8.45	8.96	9.68
		' hi'	0.03	0.02	0.02
Qwen2.5-14B	11696	dot	23.7	177.4	267.0
		white space	0.00	0.00	0.00

285, and 3,960 in Llama-3.1-8B, Qwen2.5-7B, and Qwen2.5-14B. *Random noise* adds a Gaussian random vector to the MLP output at each filler position, scaled to the size of the change caused by replacing the whole MLP output at that position; it also uses two fixed draws. For the random controls, we average correctness over the two draws within each example before bootstrap resampling.

Complete results. Table 18 lists the accuracies shown in Figure 7.

Table 18: **Accuracy after neuron-level replacement.** Accuracy (%) of the target condition, with the filler pairs of Table 9.

		Llama-3.1-8B		Qwen2.5-7B		Qwen2.5-14B	
Intervention	k	Repair	Damage	Repair	Damage	Repair	Damage
Before replacement	–	12	100	48	98	42	100
Top-k	1	94	16	98	96	74	42
	4	90	14	98	98	100	22
	16	86	16	96	98	100	26
	64	100	22	96	98	100	24
Random k	1	13	100	48	98	38	100
	4	12	100	47	98	38	100
	16	12	100	48	98	37	100
	64	12	100	48	98	39	100
Norm-matched random k	1	8	100	23	98	2	4
	4	8	100	67	95	1	4
	16	5	100	59	94	5	6
	64	8	100	61	90	5	1
Random noise	–	13	98	17	87	42	99
All neurons	d_{ff}	94	8	96	34	100	22

Agreement with the whole-MLP replacement. The selected neurons and the whole-MLP replacement also agree on individual examples. In repair, replacing the top-k neurons ($k = 1$, 1, and 4) gives the same answer as replacing the whole MLP output on 46, 49, and 50 of the 50 examples in Llama-3.1-8B, Qwen2.5-7B, and Qwen2.5-14B. In damage, the two agree on correctness for 46 examples in Llama-3.1-8B and 48 in Qwen2.5-14B.

Internal read-outs. Table 19 reports the read-outs of Table 15 for the top-k replacements. In all three models, repair lowers the readout position's attention to the filler and the final-layer norm ratio, as replacing the whole MLP output does. Damage raises both in Llama-3.1-8B and Qwen2.5-14B; in Qwen2.5-7B, where damage leaves accuracy at 96%, both remain nearly unchanged. In Qwen2.5-14B, the MLP output itself is unusually large under dot filler: its largest norm at the filler positions reaches 379 times its median over the first 500 filler positions. Replacing the four neurons lowers this ratio to 3.0 in repair and raises it from 1.7 to 599 in damage.

Table 19: **Internal read-outs of the top-k replacements.** Entries give values before $\rightarrow$ after replacing the top-1 neuron in Llama-3.1-8B and Qwen2.5-7B and the top-4 neurons in Qwen2.5-14B. *Readout attention* and *Norm ratio* follow Table 15.

Model	Source $\rightarrow$ target	Accuracy	Readout attention	Norm ratio
Llama-3.1-8B	`' hi'` $\rightarrow$ dot	12 $\rightarrow$ 94	92 $\rightarrow$ 13	4.7 $\rightarrow$ 1.2
	dot $\rightarrow$ `' hi'`	100 $\rightarrow$ 16	3 $\rightarrow$ 72	1.4 $\rightarrow$ 4.5
Qwen2.5-7B	`' hi'` $\rightarrow$ white space	48 $\rightarrow$ 98	59 $\rightarrow$ 3	2.8 $\rightarrow$ 1.3
	white space $\rightarrow$ `' hi'`	98 $\rightarrow$ 96	1 $\rightarrow$ 2	1.4 $\rightarrow$ 1.4
Qwen2.5-14B	white space $\rightarrow$ dot	42 $\rightarrow$ 100	86 $\rightarrow$ 10	3.9 $\rightarrow$ 1.5
	dot $\rightarrow$ white space	100 $\rightarrow$ 22	8 $\rightarrow$ 87	1.4 $\rightarrow$ 4.7