How Language Models Compute Negation: A Mechanistic Study of Factual Negation

Jongwook Yoon [1] **Jongwon Lim** [1] **Yohan Jo** [1]

Abstract

Large language models (LLMs) often fail to handle negation, predicting the original object of a factual statement even when a negation cue gets inserted. We study how LLMs mechanistically compute factual negation through causal interventions and logit-space analyses on pairs of original and negated factual prompts. Our analysis on Gemma-3 and Qwen3 suggests that negation is not computed as a standalone late switch that flips the original object at the final prediction position. Instead, negation is first integrated into relation information and then propagated to the final position in middle-to-late layers. In later layers, this information shifts the model away from the original object, with a notable mechanism being reducing the model's attention to subject tokens. Together, these findings suggest that LLMs' negation handling is fragile not because they have limited access to negation signals, but more likely because negation must be integrated through multi-stage modifications within a factual recall pathway.

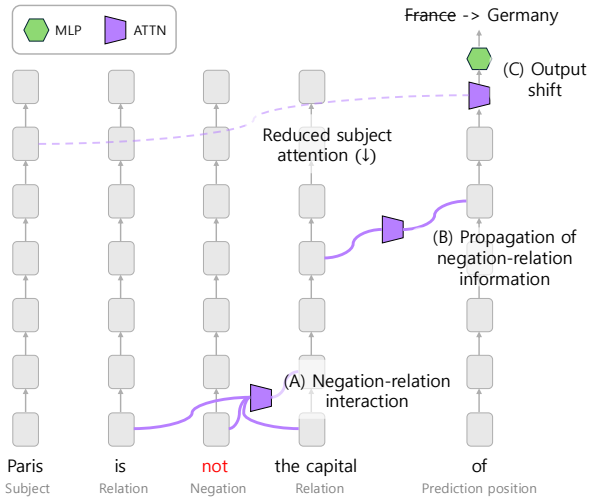


Figure 1. Overview of the proposed mechanism for factual negation. (A) Negation interacts with relation tokens in early layers. (B) Negation–relation information is propagated to the final prediction position in middle-to-late layers. (C) Later attention and MLP components shift the output away from the original object, while reduced subject attention may weaken subject-derived support.

1. Introduction

Recent large language models (LLMs) have demonstrated remarkable performance on complex reasoning and professional benchmarks, and are now being applied across a wide range of domains (OpenAI, 2023; Guha et al., 2023; Singhal et al., 2023). Despite their strong overall capabilities, LLMs continue to struggle with negation (Truong et al., 2023; García-Ferrero et al., 2023; Zhang et al., 2023). For example, when the prompt is changed from "Paris is the capital of" to "Paris is not the capital of", a model may still predict "France". These failures are prevalent according to our analysis (Table 1): among examples answered correctly under the original prompt, adding negation changes open-ended predictions in only 23–40% of cases.

To address this issue, existing work has introduced benchmarks for evaluating negation understanding (So et al., 2026), trained models on additional negated examples (García-Ferrero et al., 2023), and proposed prompt-based interventions (Barreto & Jana, 2025). However, these approaches have shown only limited improvements, suggesting that negation errors cannot be fully addressed by adding more training examples or modifying prompts alone. One reason is that such methods operate at the input-output level, without a deep understanding of how the failure arises within the model's internal computation.

To better understand the source of these failures, we analyze the internal mechanisms by which LLMs process negation. We focus on factual statements because they provide a controlled yet representative setting for studying how negation modifies factual recall. Because factual statements natu-

[1]Graduate School of Data Science, Seoul National University. Correspondence to: Jongwook Yoon <jonguki@snu.ac.kr>, Yohan Jo <yohan.jo@snu.ac.kr>.

Proceedings of the 43rd International Conference on Machine Learning, Seoul, South Korea. PMLR 306, 2026. Copyright 2026 by the author(s).

Table 1. Original-correct examples and prediction changes under negation.

Model	Setting	Original correct	Negated changed	Change (%)
CounterFact				
Gemma-3-12b-pt	Open-ended	4,633	1,861	40.2
Qwen3-8B-Base	Open-ended	3,876	1,401	36.1
Gemma-3-12b-it	Open-ended	9,266	2,733	29.5
	MCQA	17,715	15,090	85.2
Qwen3-8B	Open-ended	7,096	1,662	23.4
	MCQA	17,839	4,586	25.7
ZSRE				
Gemma-3-12b-it	Open-ended	4,133	1,123	27.2
	MCQA	10,447	10,033	96.0
Qwen3-8B	Open-ended	4,978	1,184	23.8
	MCQA	10,653	8,725	81.9

rally decompose into a subject, a relation, and an object, inserting only "not" (e.g. Paris is the capital of → Paris is "not" the capital of) lets us isolate the effect of negation on factual recall while keeping the rest of the input fixed. An intuitive possibility is that the model may first retrieve the answer suggested by the subject–relation pair, and then use the negation cue as a late signal to suppress or revise that prediction. We test this late-switch hypothesis with attention knockout (Geva et al., 2023) and activation patching (Meng et al., 2022). Next, to characterize how computations within those components drive the corresponding shifts in output logits, we combine direct logit attribution (Elhage et al., 2021) and attention-weight analysis.

Our results provide little support for the late-switch hypothesis. As summarized in Figure 1, direct attention from the final prediction position to the negation token is not the pathway through which negation affects the prediction. Instead, negation interacts with relation tokens in early layers, and the resulting information is transmitted to the final position in middle-to-late layers. Later attention and MLP components then shift the model's preference away from the original object by reducing the model's attention to subject tokens as a notable mechanism. This supports a plausible hypothesis that negation handling mainly consists of gathering less information from the subject, rather than recalling a fact and flipping it. This suggests that factual negation is difficult for LLMs not because they simply fail to detect the negation cue, but because the cue needs to be integrated into the factual recall pathway through multiple steps before the answer is selected.

2. Preliminaries

Factual recall in language models. We study negation in factual recall, where a prompt can be decomposed into a subject, relation, and object. For example, in the prompt "Paris is the capital of", the subject is "Paris", the relation is expressed by the phrase "is the capital of", and the expected object is "France". This formulation follows Geva et al. (2023), who analyze factual recall as a staged information-flow process from a subject–relation query. They show that relation information is propagated to the final prediction position, while the last subject token builds an enriched subject representation that later attention heads use to retrieve the object matching the relation-conditioned query.

Negated factual recall. We extend factual recall to minimal pairs. We call the non-negated factual prompt, such as "Paris is the capital of", the *original prompt*, and construct its *negated prompt* by inserting "not", yielding "Paris is not the capital of". The two prompts share the same subject and relation, differing only in the negation cue. This pairing provides a precise way to examine how inserting negation changes the internal factual-recall process.

3. Experimental Setup

3.1. Data

We use CounterFact (Meng et al., 2022) and ZSRE (Levy et al., 2017), constructing negated counterparts with minimal edits to the original prompt or question. For both datasets, prompts are negated by inserting "not" into the original prompt template whenever possible. After filtering, the datasets contain 21,919 CounterFact and 18,719 ZSRE examples; construction details are in § A.

3.2. Models

We analyze four recent decoder-only language models from two families: Gemma-3-12b-pt and Gemma-3-12b-it (Gemma Team, 2025), and Qwen3-8B-Base and Qwen3-8B (Qwen Team, 2025). Base models are evaluated with open-ended completion, while instruction-tuned models are evaluated in both open-ended chat and multiple-choice question answering (MCQA) formats to check whether the results persist across prompting formats. Prompting templates and MCQA details are provided in § B.

Constructing the diagnostic subset. Because our goal is to analyze how negation modifies a successful factual recall process, we first retain factual statements where the model correctly predicts the original object for the original prompt. We then measure how often this prediction changes when the corresponding negated prompt is provided to the model. Summary statistics are reported in Table 1. Prediction changes are often higher in MCQA. We speculate that this is because the model explicitly sees alternative objects, which allows the model to shift away from the original object more easily than in open-ended generation.

4. Negation Interacts with Relation Information in Early Layers

We first ask how negation information reaches the final prediction position during factual recall. Rather than acting as a late correction after the object associated with the subject-relation information has been retrieved, our results suggest that negation first interacts with relation tokens in early layers and that the resulting negation-relation information is then propagated to the final position. This section tests this pathway using attention knockout and activation patching.

4.1. Attention Knockout

To identify which attention pathways within the negated computation are necessary for the negation cue to affect the final prediction, we perform **attention knockout** with a sliding window over layers. For each negated prompt, we label tokens according to their roles: subject tokens S, relation tokens R, the negation token N, and the final prediction position F. We selectively mask one of three pathways while leaving the rest of the computation unchanged: the **negation-to-final knockout** masks attention from F to N, the **negation-subject knockout** masks attention between N and S, and the **negation-relation knockout** masks attention between N and R. In the two internal-interaction conditions, negation-subject and negation-relation knockout, we do not mask attention from F; thus, the final position can still attend to S, N, and R. We use window sizes of 7 layers for Qwen models and 11 layers for Gemma models, following prior work with windows covering roughly one-fifth of the model's layers (Geva et al., 2023).

We evaluate interventions by tracking the original object. We report the **original object prediction rate**, defined as the fraction of examples in which the model's top-1 next-token prediction is the original object. If an intervention increases this rate under the negated prompt, we interpret this as evidence that the intervened component originally plays an important role in conveying the meaning of negation. We compute this metric on examples where the original prompt predicts the original object but the corresponding negated prompt changes the output. Additional implementation details are provided in § C.

Results. We run the attention knockout across all models in § 3.2. Figure 2 shows the Qwen3-8B results in both open-ended and MCQA settings, and the remaining model- and setting-wise results are provided in § D. For both open-ended and MCQA settings, the negation–relation knockout produces the largest restoration of the original object in the early layers. This effect is strongest when the intervention window covers the early layers, and gradually decreases as the window moves to later layers. By contrast, the negation-to-final and negation–subject knockouts have a much weaker effect across all layers. This pattern suggests that negation does not affect the prediction as a direct signal read from the final position; instead, its effect depends on earlier interactions with the relation tokens.

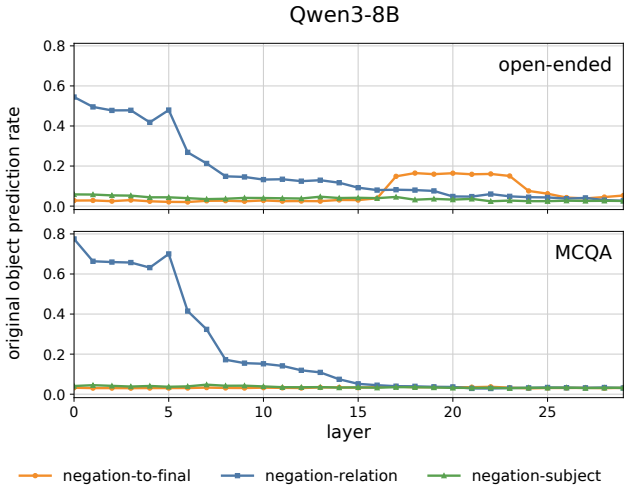


Figure 2. Layer-wise attention knockout results for Qwen3-8B in open-ended and MCQA settings, with window size 7.

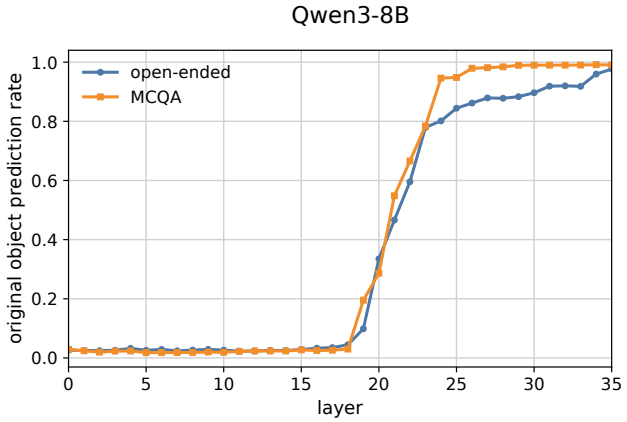


Figure 3. Layer-wise activation patching results for Qwen3-8B in open-ended and MCQA settings.

4.2. Activation Patching

To determine at which layer negation–relation information reaches the final prediction position, we perform **activation patching** between each original and negated prompt pair. For each layer l, we cache the final-position hidden state from the original prompt, replace the corresponding hidden state in the negated run, and continue the forward pass to compute the next-token distribution. If patching at layer l does not increase the original object prediction rate, this means that negation information is likely introduced in subsequent layers; otherwise, the last-position hidden state at layer l in the negated run is likely to contain negation information that contributes to final prediction. Additional

implementation details are provided in § C.

Results. The results are shown in Figure 3. For both open-ended and MCQA settings, activation patching has almost no effect in the early layers, remains near zero until the middle layers, and then rises sharply beginning around layer 20. Taken together with the attention-knockout results, this negation–relation information is propagated to the final prediction position through 18–25 layers.

5. Output Suppression Emerges in Later Layers

Having shown that negation–relation information reaches the final prediction position, we next ask how it changes the model's output preference. To identify where the preference shift from the original object to the negated prediction is produced, we first use direct logit attribution. This analysis shows that the shift is concentrated in later layers and involves both attention and MLP components. To examine what changes in information flow may accompany this shift, we then analyze final-token attention weights over the subject and relation tokens. We find that negation reduces attention to subject tokens, suggesting a possible mechanism by which support for the original object is weakened.

5.1. Direct Logit Attribution

Let x and x' be an original prompt and its negated counterpart, and let O and O' be the model's predictions under each prompt. For cases where negation changes the prediction, the model ranks O above O' under x, but reverses this order under x'. We measure this preference reversal as

$$m = \big(\log p_x(O) - \log p_x(O')\big) + \big(\log p_{x'}(O') - \log p_{x'}(O)\big).$$

Because the softmax normalizer cancels within each term, m is equivalent to a logit margin.

Let $z_x^c(o)$ denote the direct logit contribution for object o added to the final-token residual stream by component c, such as the input embedding, an attention layer, or an MLP layer. Since the final logit is a linear sum of the individual contributions of each component (Elhage et al., 2021), the contribution of component c to the preference-reversal margin is

$$m^c = z_x^c(O) - z_x^c(O') + z_{x'}^c(O') - z_{x'}^c(O).$$

A positive m^c means that component c supports the reversal from O to O', while a negative value means that it opposes the reversal. Additional details are provided in § E.

Results. The full results of direct logit attribution are summarized in § E. Across models, datasets, and prompting formats, the largest positive contributions are concentrated in middle-to-late and late layers, and involve both attention and MLP components.

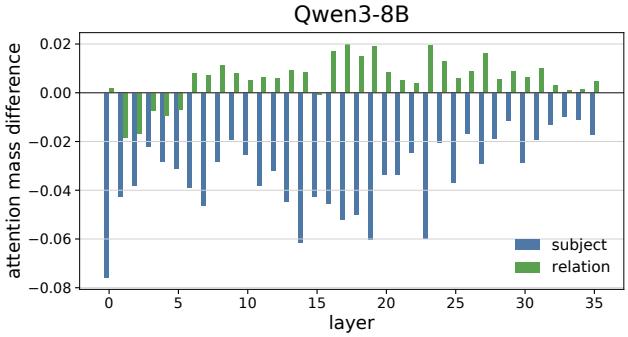


Figure 4. Layer-wise differences in final-token attention mass between negated and original prompts for Qwen3-8B.

5.2. Attention Pattern Analysis

Having identified a later-layer logit shift between the original and competing object, we next ask what factors may underlie it. To investigate this, we compare final-token attention patterns between original and negated prompts. Specifically, we compare final-token attention to subject and relation tokens by summing attention weights over each group and measuring the negated-to-original delta.

Results. As shown in Figure 4, negation consistently reduces attention to the subject tokens, while the relation-token attention difference becomes mostly positive from the middle layers onward. This indicates that negation does not simply reduce attention to all prompt components, but reallocates final-token attention away from subject information and toward the relation context. Prior work on factual recall suggests that subject information is enriched and then routed to the final prediction position to support object prediction (Geva et al., 2023). From this perspective, reduced final-token attention to the subject under negation may weaken subject-derived support for the original object, suggesting that gathering less information from the subject is a potential brittle mechanism for the logit suppression above.

6. Conclusion

Our results show that the negation cue first interacts with relation information in early layers, is propagated to the final prediction position in middle-to-late layers, and later contributes to suppressing the original object. We further find that this suppression is accompanied by reduced final-token attention to subject tokens. Overall, these findings characterize factual negation as a staged modification of the factual-recall pathway, offering a mechanistic account of how negation is processed in current language models and why this process may be fragile.

References

Barreto, J. J. D. and Jana, A. This is not a disimprovement: Improving negation reasoning in large language models via prompt engineering. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 14149–14156. Association for Computational Linguistics, 2025. URL https://aclanthology.org/2025.findings-emnlp.761/.

Elhage, N., Nanda, N., Olsson, C., Henighan, T., Joseph, N., Mann, B., Askell, A., Bai, Y., Chen, A., Conerly, T., DasSarma, N., Drain, D., Ganguli, D., Hatfield-Dodds, Z., Hernandez, D., Jones, A., Kernion, J., Lovitt, L., Ndousse, K., Amodei, D., Brown, T., Clark, J., Kaplan, J., McCandlish, S., and Olah, C. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 2021. URL https://transformer-circuits.pub/2021/framework/index.html.

García-Ferrero, I., Altuna, B., Alvez, J., Gonzalez-Dios, I., and Rigau, G. This is not a dataset: A large negation benchmark to challenge large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 8596–8615, Singapore, 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.531. URL https://aclanthology.org/2023.emnlp-main.531/.

Gemma Team. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025.

Geva, M., Bastings, J., Filippova, K., and Globerson, A. Dissecting recall of factual associations in auto-regressive language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 12216–12235, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.751. URL https://aclanthology.org/2023.emnlp-main.751/.

Guha, N., Nyarko, J., Ho, D., Ré, C., et al. Legalbench: A collaboratively built benchmark for measuring legal reasoning in large language models. In *Advances in Neural Information Processing Systems 36 (Datasets and Benchmarks Track)*, 2023.

Levy, O., Seo, M., Choi, E., and Zettlemoyer, L. Zero-shot relation extraction via reading comprehension. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pp. 333–342, Vancouver, Canada, 2017. Association for Computational Linguistics. doi: 10.18653/v1/K17-1034. URL https://aclanthology.org/K17-1034/.

Meng, K., Sharma, A., Andonian, A., Belinkov, Y., Wang, D., and Bau, D. Locating and editing factual associations in GPT. In *Advances in Neural Information Processing Systems 35*, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/6f1d43d5a82a37e89b0665b33bf3a182-Abstract-Conference.html.

OpenAI. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Qwen Team. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Singhal, K., Azizi, S., Tu, T., et al. Large language models encode clinical knowledge. *Nature*, 620(7972):172–180, 2023. doi: 10.1038/s41586-023-06291-2.

So, Y., Lee, G., Jung, S., Lee, J., Kang, J., Kim, S., and Lee, J. Thunder-NUBench: A benchmark for LLMs' sentence-level negation understanding. In *Findings of the Association for Computational Linguistics: EACL 2026*, pp. 4749–4793, Rabat, Morocco, 2026. Association for Computational Linguistics. doi: 10.18653/v1/2026.findings-eacl.250. URL https://aclanthology.org/2026.findings-eacl.250/.

Truong, T. H., Baldwin, T., Verspoor, K., and Cohn, T. Language models are not naysayers: An analysis of language models on negation benchmarks. In *Proceedings of the 12th Joint Conference on Lexical and Computational Semantics (*SEM 2023)*, pp. 101–114, Toronto, Canada, 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.starsem-1.10. URL https://aclanthology.org/2023.starsem-1.10/.

Zhang, B. and Sennrich, R. Root mean square layer normalization. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/1e8a19426224ca89e83cef47f1e7f53b-Paper.pdf.

Zhang, Y., Yasunaga, M., Zhou, Z., HaoChen, J. Z., Zou, J., Liang, P., and Yeung, S. Beyond positive scaling: How negation impacts scaling trends of language models. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 7479–7498, Toronto, Canada, 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.472. URL https://aclanthology.org/2023.findings-acl.472/.

A. Dataset Construction and Filtering

A.1. CounterFact

We use the CounterFact dataset released with ROME (Meng et al., 2022).

In most cases, the original and negated prompts differ only by the insertion of the negation token "not". For example, the original template "The mother tongue of {} is" is converted into "The mother tongue of {} is not".

We also derive multiple-choice variants of the datasets for the MCQA experiments. The construction of answer choices and the prompting templates used for MCQA are described in § B.

After filtering, the final CounterFact split contains 21,919 examples.

A.2. ZSRE

We use the ZSRE dataset released through the ROME data source (Meng et al., 2022).[1] Unlike CounterFact, ZSRE does not provide explicit relation identifiers. Each example instead contains a natural-language question in the `src` field, along with a subject and an object. Therefore, we construct negated inputs at the question-template level rather than at the relation-id level.

For each question template, we define a negated template that preserves the original question structure as much as possible. When possible, the negated version is created by inserting only "not". However, some question templates contain active verbs for which simply inserting "not" would be ungrammatical. In such cases, we apply a minimal grammatical edit using do-support. For example, "Who developed {}?" is converted into "Who did not develop {}?" Templates that could not be naturally negated with a minimal edit were excluded.

The original ZSRE evaluation split contains 19,086 examples. After removing templates that could not be minimally negated, the final ZSRE split contains 18,719 examples. Among these, 16,980 examples are negated by inserting only "not", and 1,739 examples require a minimal do-support verb edit. At the template level, the original split contains 892 question templates; we construct negated versions for 871 templates and skip 21 templates. Because ZSRE consists of question-style inputs, open-ended completion with base models rarely produces the answer directly; we therefore evaluate ZSRE only with instruction-tuned models.

B. Prompting and MCQA Construction

This appendix describes the prompting formats used for base and instruction-tuned models, as well as the construction of the MCQA setting.

B.1. Open-ended Completion for Base Models

For base models, we use the factual prompt directly in an open-ended completion format. Given an original or negated prompt, the model is asked to continue the prompt, and we evaluate the probability assigned to the original object.

B.2. Open-ended Format for Instruction-tuned Models

For instruction-tuned models, we wrap the same open-ended factual completion task in a chat-style instruction format. The system prompt instructs the model to complete the user's text with the single most likely continuation and to output only the answer string.

The prompt format is as follows:

> **System:** Complete the user's text with the single most likely next word based on your knowledge about the world. Output only that word, with no punctuation, no quotation marks, and no explanation.
>
> **User:** Paris is the capital city of
>
> **Assistant:**

[1] https://rome.baulab.info/data/dsets/zsre_mend_eval.json

This format allows instruction-tuned models to be evaluated in a setting that is close to the open-ended completion format used for base models, while respecting their chat-based training format.

B.3. MCQA Format for Instruction-tuned Models

We additionally evaluate instruction-tuned models in a multiple-choice question answering format. This setting reduces the possibility that our findings are specific to open-ended generation or to the surface form of original-object completion. In MCQA, the model receives a prompt and four answer choices, and is instructed to output only the choice label.

The prompt format is as follows:

> **System:** Answer the multiple-choice question with only the correct choice label. Do not include punctuation, explanation, or any other text.
>
> **User:** Complete the following prompt by choosing the correct option.
>
> Prompt: The mother tongue of Danielle Darrieux is
>
> A. Turkish
>
> B. French
>
> C. Spanish
>
> D. Hungarian
>
> Answer with only the choice label.
>
> **Assistant:**

We evaluate the model by comparing its preference over the four choice labels.

B.4. MCQA Construction

For each MCQA example, we construct four answer choices consisting of the original object and three distractors. For CounterFact, distractors are sampled from the pool of `target_true` and `target_new` objects associated with the same `relation_id`. For ZSRE, which does not provide explicit relation identifiers, distractors are sampled from original objects associated with the same question template.

We filter out distractors with high lexical overlap with the correct answer to avoid trivially similar answer choices. To reduce option-position bias, we create separate versions of the MCQA dataset in which each of A, B, C, and D is used as the correct answer position.

C. Intervention Details

Token indexing. Subject and negation-token positions are obtained by matching their character spans in the prompt and mapping the matched spans to token positions using the tokenizer offset mapping. The final prediction position is defined as the last token position of the input sequence. Relation-context positions are implemented as all token positions within the factual prompt span that occur before the final prediction position and are not assigned to either the subject or the negation token. For instruction-tuned models, we identify this span inside the user message. In the MCQA setting, this means that answer choices and surrounding multiple-choice instructions are excluded from the relation-context positions.

Attention-mask intervention. For attention knockout, we modify the attention mask inside the selected self-attention modules. Given a blocked query–key pair (i, j), we clone the attention mask and set the entry corresponding to query position i and key position j to `torch.finfo(mask.dtype).min`. This makes the selected attention edge unavailable while leaving the rest of the forward pass unchanged. After the intervention run, the original self-attention forward functions are restored.

For final-position knockout, the query position is fixed to the final prediction position, and the blocked key positions are the selected token group. For internal-interaction knockout, we construct all causally valid cross-group pairs between two token groups. Specifically, for groups A and B, we include (i, j) for $i \in A, j \in B$ when $j \leq i$, and symmetrically include (i, j) for $i \in B, j \in A$ when $j \leq i$. Duplicate pairs are removed before applying the mask.

Layer windows. Attention knockout is applied over consecutive layer windows. For a window starting at layer l with size w, the intervention is applied to layers

$$l, l+1, \ldots, l+w-1.$$

The window size is read from the model registry unless explicitly overridden by the command-line argument.

Activation patching. For activation patching, we first run both prompts in the pair and cache the output hidden state at the final prediction position for every layer. At layer l, we register a forward hook on the corresponding transformer block. During the target forward pass, the hook clones the layer output and replaces only the final-position vector with the cached source vector from the same layer:

$$h^{(l)}_{\text{target},F} \leftarrow h^{(l)}_{\text{source},F}.$$

All other token positions and all subsequent layers are left unchanged. The main experiments use the original prompt as the source and the negated prompt as the target.

Saved outputs. For each layer or layer window, we save the top-1 prediction after intervention, the probability assigned to the original-object token, the change in original-object probability relative to the corresponding baseline, and the rate at which the intervention changes the top-1 prediction from the unmodified baseline. For attention knockout, we also save the start and end layer of each window; for internal-interaction knockout, we additionally record the mean number of blocked attention edges.

D. Additional Results for Attention Knockout and Activation Patching

This section reports additional model- and setting-wise results that are omitted from the main text for space. Across model families and evaluation settings, the qualitative pattern is consistent with the main results: masking negation–relation attention produces the strongest restoration of the original object in early layers, whereas directly masking attention from the final prediction position to the negation token has a weaker effect. Activation patching at the final prediction position has little effect in early layers but restores the original object more strongly in middle-to-late layers. The corresponding results are shown in Figures 10, 11, 12, and 13.

E. Direct Logit Attribution

E.1. Overview

Direct logit attribution (DLA) decomposes the model's preference between two candidate objects into contributions from individual residual-stream components. We use it to identify which attention and MLP components support the observed preference shift between the original and negated prompts.

For each original prompt x and its negated counterpart x', let O denote the object preferred under the original prompt and O' denote the object preferred under the negated prompt. For a component c, such as an attention output or an MLP output, let r^c_x be the vector contributed by that component to the final prediction position. We project this vector onto centered unembedding directions,

$$\bar{W}_U[:, o] = W_U[:, o] - \frac{1}{|\mathcal{V}|} \sum_{v \in \mathcal{V}} W_U[:, v],$$

and define the component's direct contribution to object o as

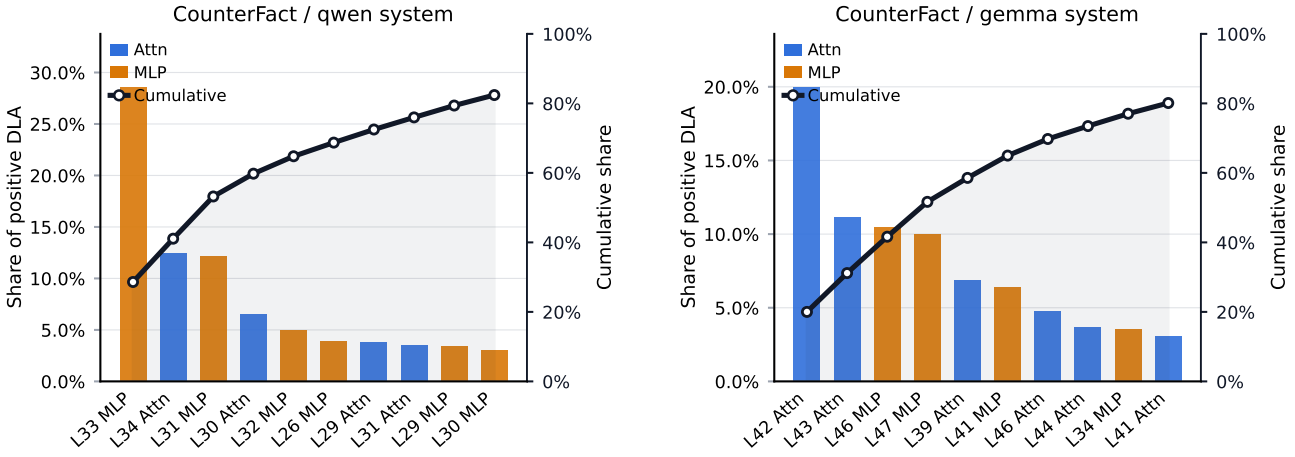
$$z^c_x(o) = \left\langle r^c_x, \bar{W}_U[:, o] \right\rangle.$$

Centering removes token-independent logit shifts, which do not affect the relative preference between candidate objects.

The component contribution to the original–negated preference reversal is then

$$m^c = \big(z^c_x(O) - z^c_x(O')\big) + \big(z^c_{x'}(O') - z^c_{x'}(O)\big).$$

A positive value means that component c supports the observed reversal from O under the original prompt to O' under the negated prompt. A negative value means that the component opposes this reversal. In the figures below, we rank components by their positive contribution and report the fraction of the total positive DLA mass explained by the top components.



(a) Qwen3-8B, open-ended *(b)* Gemma-3-12b-it, open-ended

Figure 5. Top positive direct logit attribution components on CounterFact in the open-ended setting. Bars show the share of total positive DLA mass explained by each component, and the line shows the cumulative share.

E.2. Accounting for RMSNorm

All models analyzed in this work use RMSNorm (Zhang & Sennrich, 2019) before the final unembedding layer. RMSNorm introduces a subtlety for direct logit attribution because the normalization scale depends on the full final hidden state. Therefore, the model's final readout is not globally linear in the component vectors.

We handle this by fixing the RMSNorm scale to the value computed from the observed final hidden state. Let h_x denote the final hidden state at the prediction position for prompt x, and let r_x^c denote the vector added by component c at that position. The model's final logit for object o is

$$\ell_x(o) = \left\langle \frac{\gamma \odot h_x}{\rho_x}, W_U[:, o] \right\rangle, \qquad \rho_x = \sqrt{\frac{1}{d}\|h_x\|_2^2 + \epsilon},$$

where γ is the learned RMSNorm scale and d is the hidden dimension.

For DLA, we treat ρ_x, computed from the final hidden state h_x, as fixed. Under this fixed normalization scale, the readout becomes linear in each component vector. We therefore define the direct contribution of component c to object o as

$$z_x^c(o) = \left\langle \frac{\gamma \odot r_x^c}{\rho_x}, \bar{W}_U[:, o] \right\rangle,$$

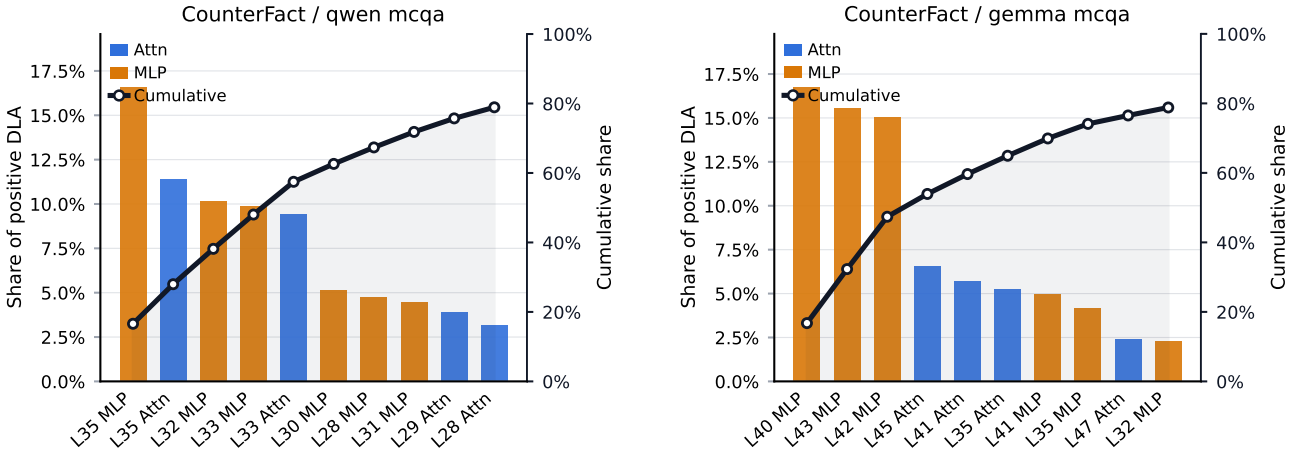
where $\bar{W}_U$ is the centered unembedding matrix defined above. Equivalently, each component is projected through the same RMSNorm scale that the model applies to the final hidden state in that forward pass.

This gives an additive decomposition of the fixed-RMSNorm linear readout: once ρ_x is fixed, the component contributions sum to the logit contribution of the full final hidden state under that same scale. However, this should not be interpreted as a counterfactual intervention. If a component were removed or changed, the final hidden state would also change, and the RMSNorm scale ρ_x could change as well. Accordingly, we use DLA as a descriptive attribution of the observed forward computation under a fixed final-state normalization scale. All prediction changes, object prediction rates, and intervention margins are computed using the model's actual post-RMSNorm logits.

E.3. Additional Experimental Results

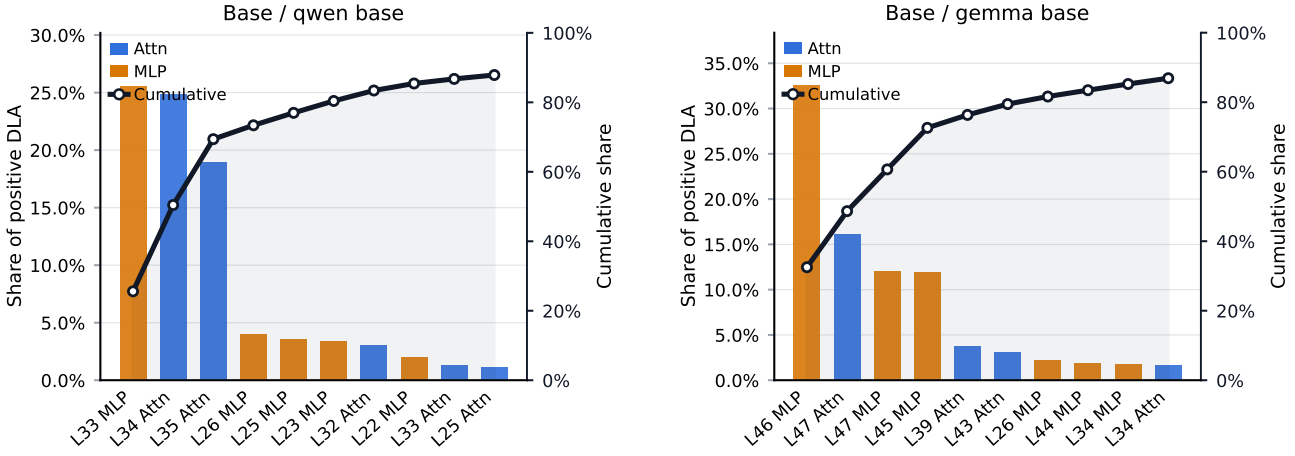
Figure 5 shows the top positive DLA components for the open-ended CounterFact setting. For both Qwen and Gemma, the largest positive contributions are concentrated in middle-to-late and late layers, and include both attention and MLP components.

Figure 6 shows the corresponding CounterFact MCQA results. The MCQA setting changes the output space from open-ended token prediction to choice-label prediction, but the qualitative pattern remains similar: a small number of later



(a) Qwen3-8B, MCQA *(b)* Gemma-3-12b-it, MCQA

Figure 6. Top positive direct logit attribution components on CounterFact in the MCQA setting. Bars show per-component share and the line shows cumulative share.



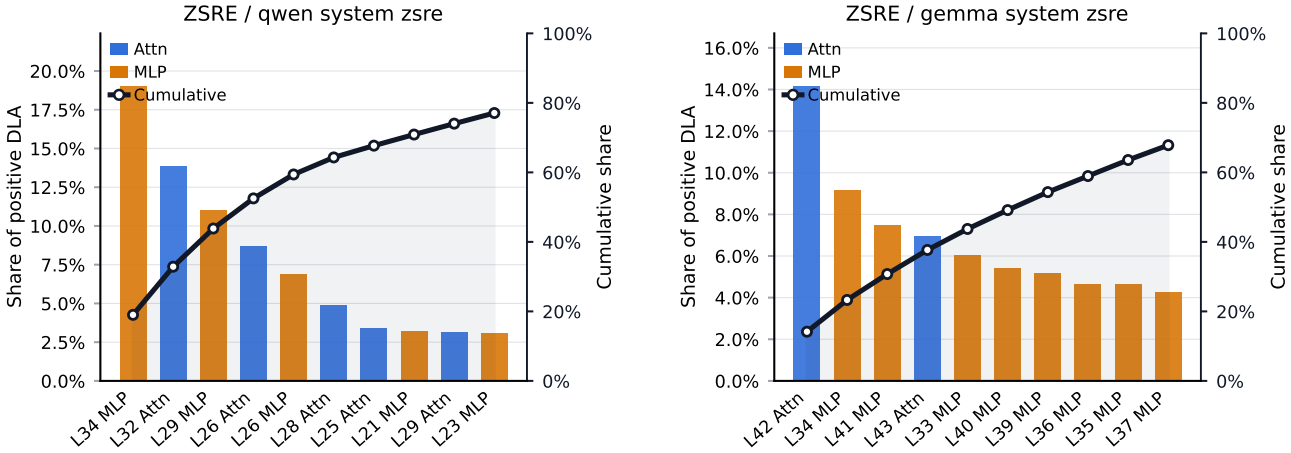
(a) Qwen3-8B-Base *(b)* Gemma-3-12b-pt

Figure 7. Top positive direct logit attribution components for base models on CounterFact in the open-ended setting.

components explain a large fraction of the positive DLA mass.

Figure 7 reports open-ended results for the base models on CounterFact. These results provide a comparison point showing that the later-component concentration is not specific to instruction tuning.

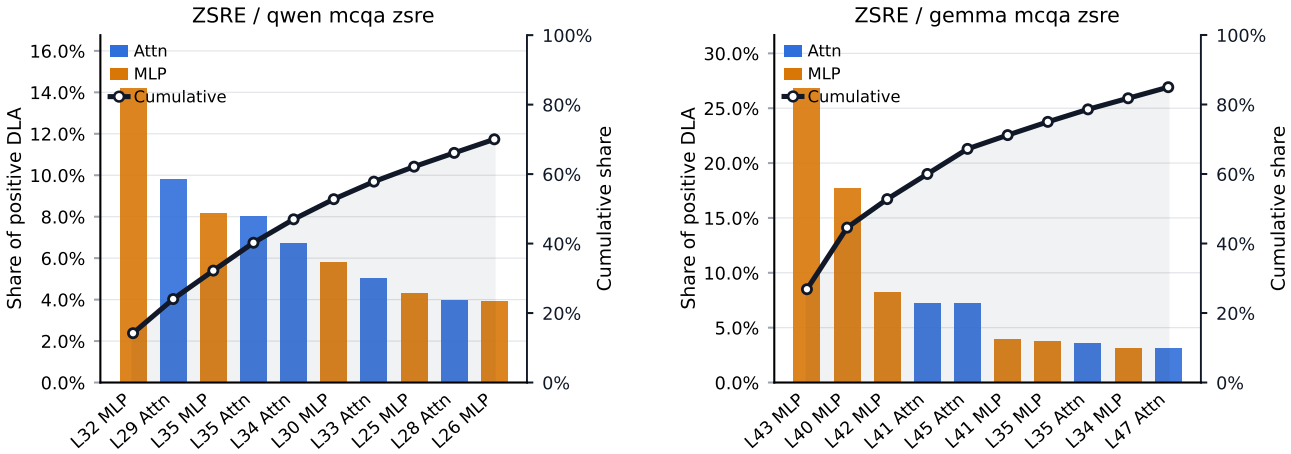
Figures 8 and 9 report the same analysis on ZSRE. Although ZSRE uses question-style prompts rather than CounterFact-style completion prompts, the main qualitative result remains: the preference shift is concentrated in a small set of middle-to-late and late components.



(a) Qwen3-8B, open-ended

(b) Gemma-3-12b-it, open-ended

Figure 8. Top positive direct logit attribution components on ZSRE in the open-ended setting.



(a) Qwen3-8B, MCQA

(b) Gemma-3-12b-it, MCQA

Figure 9. Top positive direct logit attribution components on ZSRE in the MCQA setting.

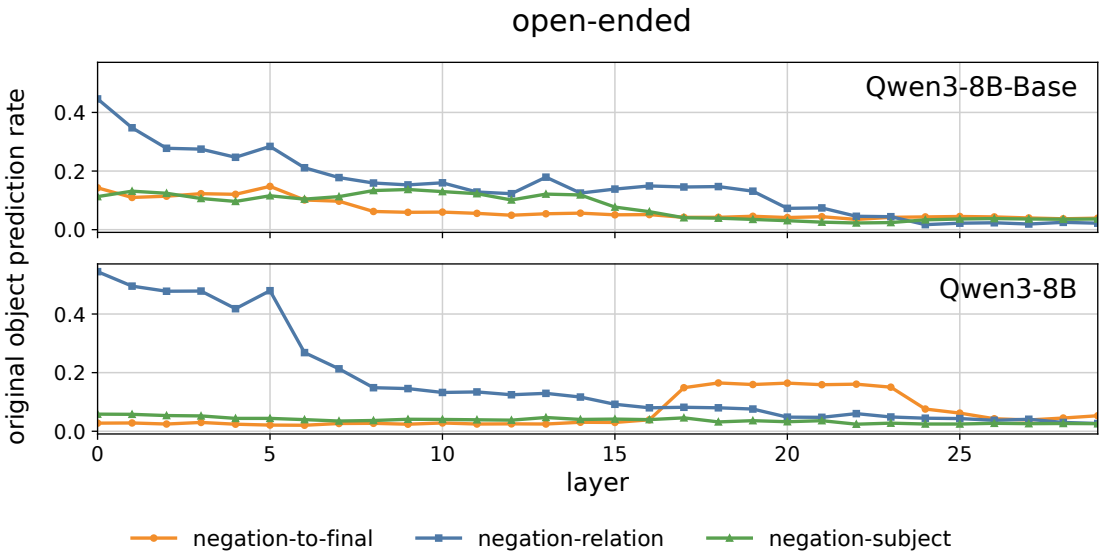


Figure 10. Additional layer-wise attention knockout results for Qwen3-8B-Base and Qwen3-8B on open-ended settings. The negation–relation knockout consistently produces the strongest early-layer restoration effect.

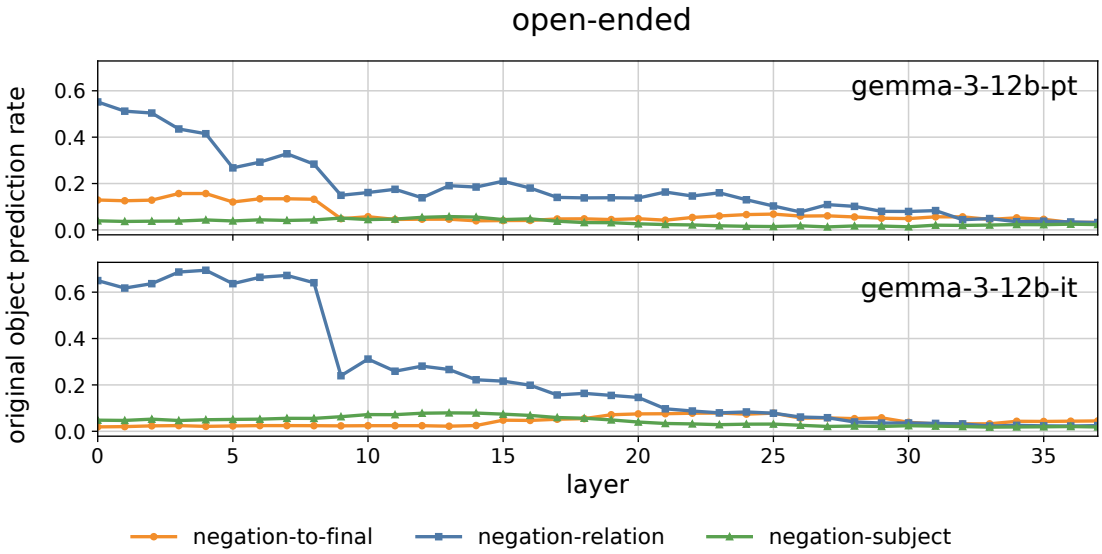


Figure 11. Additional layer-wise attention knockout results for Gemma-3-12b-pt and Gemma-3-12b-it on open-ended settings. The negation–relation knockout consistently produces the strongest early-layer restoration effect.

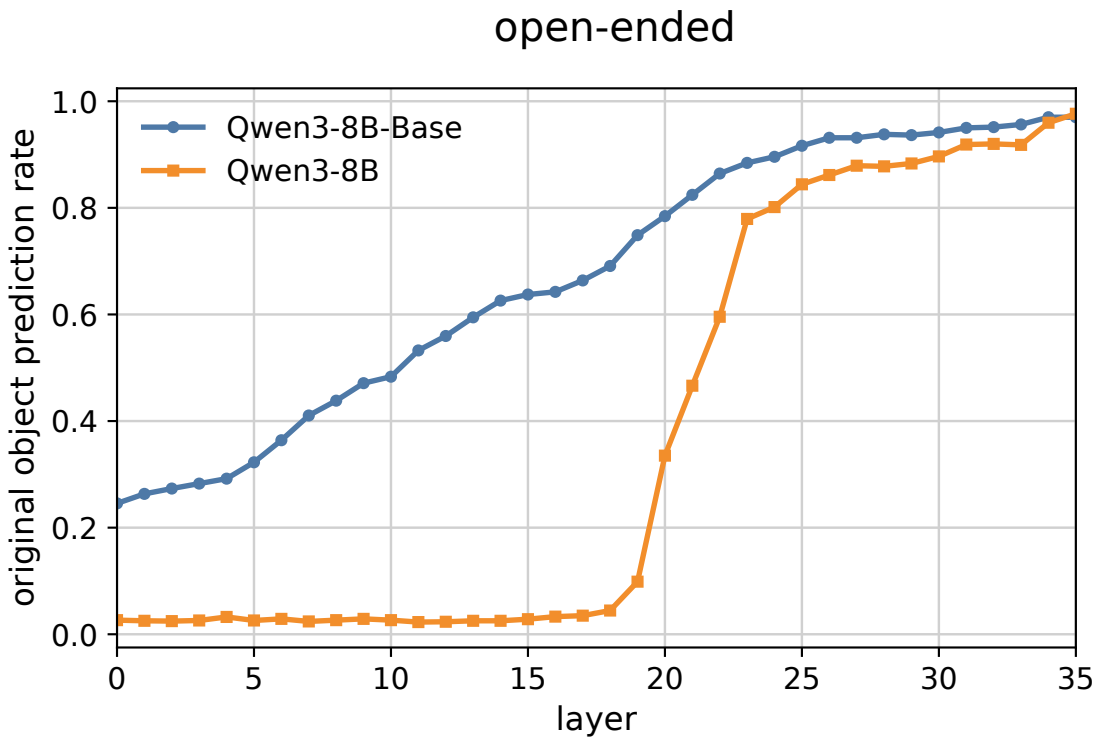


Figure 12. Additional layer-wise activation patching results for Qwen3-8B-Base and Qwen3-8B on open-ended settings.

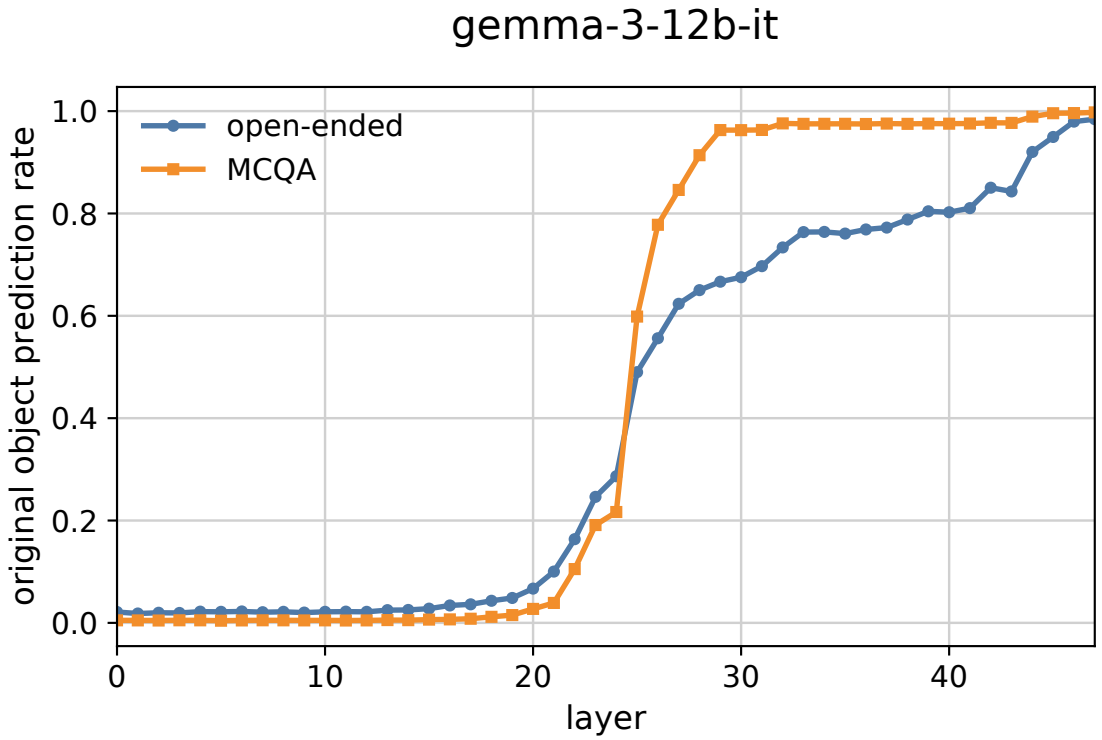


Figure 13. Additional layer-wise activation patching results for Gemma-3-12b-it in open-ended and MCQA settings.