Towards a Mechanistic Understanding of Propositional Logical Reasoning in Large Language Models

Anonymous ACL submission

Abstract

Understanding how Large Language Models (LLMs) perform logical reasoning internally remains a fundamental challenge. While prior mechanistic studies focus on identifying task-specific circuits, they leave open the question of what computational strategies LLMs employ for propositional reasoning. We address this gap through comprehensive analysis of *Qwen3* (8B and 14B) on *PropLogic-MI*, a controlled dataset spanning 11 propositional logic rule categories across one-hop and two-hop reasoning. Rather than asking "which components are necessary," we ask "how does the model organize computation?" Our analysis reveals a coherent computational architecture comprising four interlocking mechanisms: Staged Computation (layer-wise processing phases), Information Transmission (information flow aggregation at boundary tokens), Fact Retrospection (persistent re-access of source facts), and Specialized Attention Heads (functionally distinct head types). These mechanisms generalize across model scales, rule types, and reasoning depths, providing mechanistic evidence that LLMs employ structured computational strategies for logical reasoning.

1 Introduction

The capability of Large Language Models (LLMs) to execute complex logical reasoning has improved significantly (Cheng et al., 2025; OpenAI et al., 2024; Guo et al., 2025). State-of-the-art evaluations demonstrated this trend: *GPT-5.2* achieves 92% accuracy on *GPQA-Diamond*, while *Gemini 3 Pro* reaches 76% on *SimpleBench*. However, the underlying computational mechanisms behind such high empirical performance remain poorly understood. This gap in understanding raises a fundamental question: are these models simply sophisticated pattern matchers relying on surface statistics (Bender et al., 2021; Dziri et al., 2023; Zhang et al., 2022), or have they internalized robust algorithms that mirror logical deduction (Li et al., 2024; Nanda et al., 2023)? To address this ambiguity, the emerging field of *Mechanistic Interpretability* (MI) (Sharkey et al., 2025) has begun to reverse-engineer transformers to causally uncover the mechanisms underlying model behaviors. Recent efforts have applied MI to reasoning domains including arithmetic (Kantamneni and Tegmark, 2025; Stolfo et al., 2023; Yu and Ananiadou, 2024), syllogistic inference (Kim et al., 2025), and multi-hop composition (Ye et al., 2025; Xia et al., 2024). Their findings suggest that LLMs may employ structured, interpretable strategies when processing logical reasoning, rather than relying solely on opaque pattern matching.

Propositional logical reasoning, *i.e.*, the capacity to manipulate boolean operators and chain inference rules, represents a cornerstone of formal deduction that underpins more complex reasoning tasks. Despite its foundational role, this domain remains largely unexplored in MI. Behavioral studies reveal that model accuracy degrades significantly with increased proof depth (Saparov and He, 2023; Dziri et al., 2023), yet the internal computational mechanisms responsible for these failures remain opaque. A prior mechanistic work, Hong et al. (2025), provides circuit-level analysis identifying four attention head families in *Mistral-7B*, *Gemma-2-9B*, and *Gemma-2-27B*. While they demonstrate that similar circuits emerge across these models, their study is constrained to a single operator structure (*i.e.*, implication chains) using simplified one-hop problems. Broader propositional rules, including *De Morgan's Laws*, *Distributivity*, and *Commutativity*, as well as applying multiple different rules in a multi-step reasoning process, remain mechanistically unexplored. Moreover, their circuit-discovery paradigm focuses on identifying *which specific components* are necessary for a given model-task pair, rather than characterizing *how* the reasoning process is organized at a higher level

of abstraction, namely the generalizable computational strategies that may transfer across model architectures and logical structures.

In this paper, we address this gap with a more comprehensive analysis. First, we shift analytical focus from model-specific circuit identification to generalizable mechanistic principles. Rather than asking *"which heads are necessary for this specific task,"* we ask *"what computational strategies does the model employ to solve logical problems in general?"* As shown in Figure 1, this perspective reveals four interlocking mechanisms: (1) Staged Computation, with layer-wise processing phases where early, middle, and late layers assume distinct functional roles; (2) Information Transmission, where semantic information aggregates at boundary tokens; (3) Fact Retrospection, in which factual information is persistently re-accessed; and (4) Specialized Attention Heads, where attention heads exhibit stable functional specialization. Second, we expand the scope of analysis to 11 propositional logic rule categories across one-hop and two-hop reasoning in *Qwen3* (8B, 14B), demonstrating that these mechanisms persist across model scales, rule types, and reasoning depths. Through the above analysis, we provide a more profound, complete and generalizable understanding of the mechanism of propositional logical reasoning.

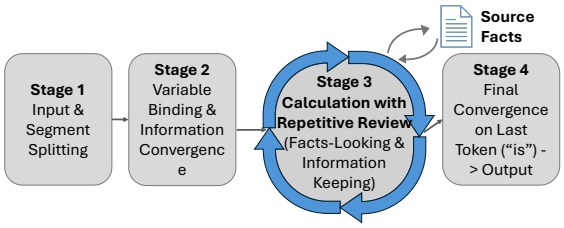
2 Related Work

Mechanistic Interpretability of Transformers. This area reverse-engineers transformer computations to understand capabilities like language modeling (Olsson et al., 2022), factual recall (Meng et al., 2023; Geva et al., 2023; Ferrando and Voita, 2024), and entity binding (Feng and Steinhardt, 2024; Wang et al., 2023). The most popular methods used in MI is causal mediation analysis, representive by patching methods (Vig et al., 2020; Hase et al., 2023; Zhang and Nanda, 2024; Heimersheim and Nanda, 2024). Circuit-based interpretability (Elhage et al., 2021; Wang et al., 2022) identifies minimal subgraphs for specific tasks, while more recent work explores distributed representations through sparse autoencoders (Marks et al., 2024; Engels et al., 2025) and function vectors (Todd et al., 2024). However, these studies primarily focus on language modeling primitives (*e.g.*, copying, entity tracking) or isolated task components, without examining end-to-end reasoning pathways in multi-step logical tasks. Our work extends mechanistic interpretability to propositional logical reasoning, tracing information flow across residual streams, attention heads, and MLPs for comprehensive logical rules across one-hop and two-hop scenarios, representing a more systematic investigation than prior component-level analyses.

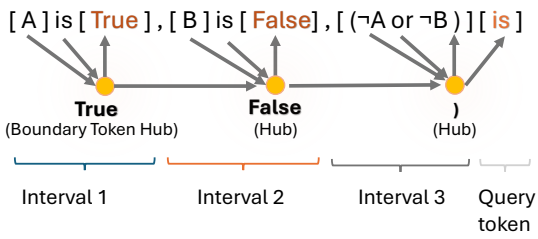
Propositional Logic Reasoning in LLMs Benchmarks such as PrOntoQA (Saparov and He, 2023), ProofWriter (Tafjord et al., 2021), and LogicBench (Parmar et al., 2024) have been developed to evaluate LLM propositional reasoning. Behavioral studies reveal systematic limitations specific to logical inference: accuracy degrades significantly with proof depth (Saparov and He, 2023; Dziri et al., 2023), models struggle with proof planning (selecting correct inference steps when multiple valid options exist (Saparov and He, 2023)) and exhibit particular difficulty with negated premises (Parmar et al., 2024). These findings fuel the ongoing debate: do LLMs implement genuine logical algorithms (Nanda et al., 2023), or merely exploit statistical shortcuts (Dziri et al., 2023)? However, behavioral evaluation alone only answers *whether* models succeed, not *how* models complete the reasoning internally. To the best of our knowledge, the only mechanistic study on propositional logic is Hong et al. (2025), which decomposes implication reasoning into four functional head types (locating, mover, processing, and decision heads) in *Mistral-7B* and *Gemma-2* models. Their component-level analysis successfully pinpoints specific heads responsible for rule retrieval, information movement, and answer selection. Different from this head-level functional decomposition, we characterize network-wide computational patterns: how processing is temporally organized across layers, how information flows across token positions , and how facts are repeatedly retrieved throughout network depth. This pattern-level perspective, combined with broader rule coverage and multi-hop scenarios, reveals mechanisms that generalize across diverse logical structures, inference chain length and model scales.

3 Methodology

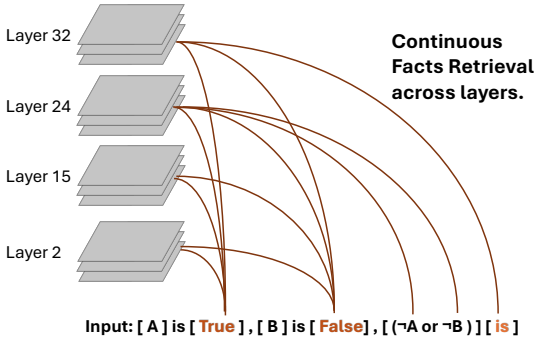
Our approach departs from the circuit-discovery paradigm (Elhage et al., 2021; Wang et al., 2022) which primarily asks "which heads are necessary?" We instead focus on "what computational strategies does the model employ?" This shift is motivated



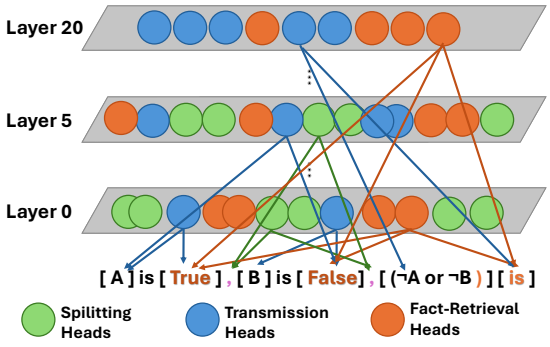
(a) **Staged Computation**



(b) **Information Transmission**



(c) **Fact Retrospection**



(d) **Specialized Attention Heads**

Figure 1: **Overview of the propositional logic reasoning mechanisms in *Qwen3*.** (a) Staged Computation: MLP patching reveals layer-wise processing phases. (b) Information Transmission: Semantic content aggregates at segment-terminal tokens. (c) Fact Retrospection: Fact tokens maintain persistent causal influence across depth. (d) Specialized Attention Heads: Specialized heads implement the macroscopic mechanisms.

by two considerations. First, circuit faithfulness is difficult to establish, as multiple circuits may implement the same function and ablating one may simply shift computation to another. Second, circuits often fail to generalize across different models, limiting their scientific value for understanding *how transformers reason* rather than *how a particular model is wired*. Accordingly, our analysis characterizes network-wide computational patterns rather than minimal circuits.

3.1 Task Formulation

We formulate the propositional logical reasoning task as a next-token prediction problem. Given a prompt $x = [F_1, \ldots, F_k, R_1, \ldots, R_m, Q]$, where $\{F_i\}$ are premises (*i.e.*, facts), $\{R_j\}$ are logical expressions, and Q is the query, the model $\mathcal{M}$ predicts the answer $y \in \{\texttt{True}, \texttt{False}\}$. To strictly evaluate implicit reasoning, we require the model to output the answer immediately following the query without *Chain-of-Thought* (CoT) generation, confining all reasoning to internal hidden states.

3.2 Dataset Construction

PropLogic-MI While benchmarks like *PrOntoQA* and *ProofWriter* have established a foundation for evaluating reasoning accuracy in LLMs, they remain insufficient for fine-grained MI. These datasets often introduce uncontrolled linguistic variability and entangled reasoning steps, which act as confounding factors for causal analysis methods like activation patching. To address this limitation, we introduce *PropLogic-MI*, a highly controlled dataset designed specifically for circuit discovery. *PropLogic-MI* focuses on atomic logical operations across 11 propositional logic rule categories (Table 1), providing strictly aligned clean/corrupted pairs for one-hop and two-hop reasoning. This structural precision enables researchers to isolate the exact neural components responsible for logical transitions, shifting the focus from measuring performance to dissecting the underlying mechanism.

Prompt Template For each rule, we instantiate prompts by enumerating all possible truth value assignments to atomic propositions. For example, for *De Morgan's Law*:

```
"A is True, B is False, (¬A or ¬B) is"
```

where the model is expected to predict the correct truth value (`True` in this case). The ground-truth answer is computed via symbolic evaluation.

Reasoning Depth We construct prompts under two complexity settings: (1) One-Hop Reasoning,

Category	Formal Definitions	Example Prompt Templates (One-Hop)
Identity	$P \wedge \top \equiv P, \quad P \vee \bot \equiv P$	`"A is T, A and T is", "A is F, A or F is"`
Domination	$P \wedge \bot \equiv \bot, \quad P \vee \top \equiv \top$	`"A is T, A and F is", "A is F, A or T is"`
Idempotent	$P \wedge P \equiv P, \quad P \vee P \equiv P$	`"A is T, A and A is", "A is F, A or A is"`
Dbl. Negation	$\neg(\neg P) \equiv P$	`"A is T, (¬(¬A)) is"`
Excluded Mid.	$P \vee \neg P \equiv \top$	`"A is T, A or ¬A is"`
Contradiction	$P \wedge \neg P \equiv \bot$	`"A is F, A and ¬A is"`
Commutative	$P \wedge Q \equiv Q \wedge P$	`"A is T, B is F, A and B is"`
Associative	$(P \wedge Q) \wedge R \equiv P \wedge (Q \wedge R)$	`"A is T, B is T, C is F, (A and B) and C is"`
Distributive	$P \wedge (Q \vee R) \equiv (P \wedge Q) \vee (P \wedge R)$	`"A is T, B is F, C is T, A and (B or C) is"`
De Morgan	$\neg(P \wedge Q) \equiv \neg P \vee \neg Q$	`"A is T, B is F, (¬A or ¬B) is"`
Absorption	$P \wedge (P \vee Q) \equiv P$	`"A is T, B is F, A and (A or B) is"`

Table 1: **Rule coverage.** *PropLogic-MI* systematically covers 11 fundamental rule categories. Each rule includes its formal Boolean algebra definition and a corresponding template. Variables {`A, B, C`} are instantiated with truth values (`True`/`False`). T/F denotes `True`/`False`.

Depth	Prompt Template & Reasoning Process
One-Hop	`"A is True, B is False, (A and B) is"` ↪ *Logic:* $A \wedge B$
	`"A is True, (¬(¬A)) is"` ↪ *Logic:* $\neg(\neg A) \equiv A$
	`"A is True, B is False, (¬A or ¬B) is"` ↪ *Logic: De Morgan* $(\neg A \vee \neg B)$
Two-Hop	`"A is T, B is A and T, C is F, B and C is"` ↪ *Chain:* $(A \wedge \top) \to B; \quad (B \wedge C) \to Ans$
	`"A is T, B is F, C is ¬(A and B), D is T, C or D is"` ↪ *Chain:* $NAND(A, B) \to C; \quad (C \vee D) \to Ans$
	`"A is T, B is F, C is A and B, D is T, C or D is"` ↪ *Chain:* $(A \wedge B) \to C; \quad (C \vee D) \to Ans$

Table 2: **Probing prompts across reasoning depths.** We probe the model with queries requiring direct rule application (One-Hop) versus those necessitating intermediate latent variable derivation (Two-Hop).

where the query is directly derived from facts via a single rule, and (2) Two-Hop Reasoning, where an intermediate conclusion must first be derived before predicting the final answer. Representative examples are provided in Table 2.

3.3 Analytical Methods

Causal Framework To rigorously trace information flow underlying logical reasoning, we move beyond correlational analysis (*e.g.*, attention maps) to establish causal sufficiency. We employ *Activation Patching*, a method grounded in counterfactual interventions, and the procedure is as follows: construct a *clean* input that yields the correct answer and a *corrupted* input that yields a different prediction; then selectively patch activations in the corrupted run with their clean counterparts. If patching a specific component recovers the correct answer, we identify it as causally critical.

Implementation and Metrics We construct paired inputs for this counterfactual setup: a *clean* input x_{clean} yielding target answer Y_{clean}, and a *corrupted* input X_{corrupt} minimally edited (*e.g.*, flipping a truth value) to induce a contrasting prediction Y_{corrupt}. We first perform a forward pass on X_{clean} to cache activations $h_{l,t}$ (or $h_{l,h,t}$ for attention head h) at layer l and token position t. And then we perform a forward pass on $X_{corrupt}$ and get $Y_{corrupt}$. We again run a forward pass on X_{corrupt}, at this time we intervene by replacing the target activation with its cached clean counterpart and get $Y_{patched}$.

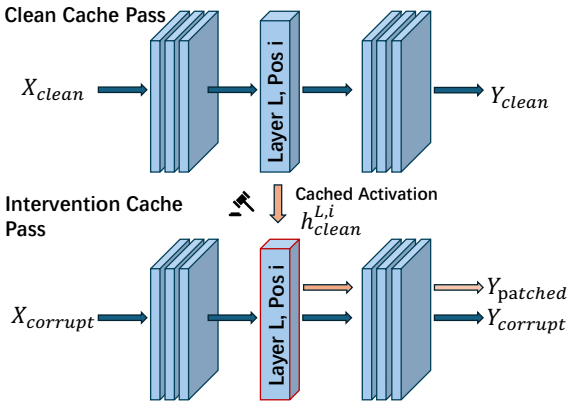


Figure 2: Activation patching procedure. We focus on the logits difference between `True` token and `False` token in $Y_{patched}$ output logits.

We quantify the causal effect of this intervention using the *Logit Difference* (LD) metric. Let $w_{\texttt{True}}$ and $w_{\texttt{False}}$ denote the tokens for the two truth values(assume that `True` is the correct answer on X_{clean}, while `False` is the correct answer on $X_{corrupt}$, normally the two are mutually exclusive).

The LD on the patched output is:

$$\text{LD} = \text{Logits}(w_{\texttt{True}}) - \text{Logits}(w_{\texttt{False}}) \quad (1)$$

A positive LD indicates that the component (l, t) (or (l, h, t)) retains critical information required to steer the prediction back to y_{clean}. To quantify the causal effect of patching, we define the *Logit Difference Shift* (dLD):

$$\text{dLD} = \text{LD}_{\text{patched}} - \text{LD}_{\text{baseline}} \quad (2)$$

where $\text{LD}_{\text{baseline}}$ is the logit difference on the unpatched corrupted run. A positive dLD indicates successful restoration toward the correct answer.

3.4 Experimental Setup

Models Our primary model is *Qwen3-8B* (36 layers, 32 attention heads per layer). To verify cross-model consistency, key experiments are replicated on *Qwen3-14B*. All analyses use pretrained checkpoints in zero-shot settings without fine-tuning.

PropLogic-MI We constructed a total of 370 samples (74 one-hop, 296 two-hop) by populating the templates in Table 1 with randomized truth values, thereby generating strictly aligned clean and corrupted pairs. To ensure the validity of our mechanistic interventions, we constraint that the model must accurately predict the clean and corrupt pair to differentiate genuine circuit disruption from pre-existing performance failures. Following evaluation with *Qwen3-8B*, we retain 224 validated examples (60.5%), comprising 42 one-hop (56.8%) and 182 two-hop (61.5%) cases. This filtered subset preserves sufficient statistical power while maintaining balanced coverage across reasoning depths and rule categories.

Evaluation We apply activation patching across all 11 rule categories (§3.2) using LD (Eq. 1) and dLD (Eq. 2) to quantify component importance. Each experiment performs three passes: (1) clean run to cache activations; (2) corrupted baseline; (3) patched run with targeted replacement. All analyses run on NVIDIA H100 GPUs using TransformerLens (Nanda and Bloom, 2022). Detailed implementation is provided in Appendix A.

4 Experiments

We present our findings in four subsections, progressing from macroscopic to microscopic analysis. Section 4.1 establishes layer-wise processing phases via MLP patching. Sections 4.2 and 4.3 trace information flow through residual stream analysis, revealing both transmission patterns and persistent premises (*i.e.*, truth values) access. Section 4.4 validates these patterns at the attention head level. Figure 1 provides a schematic overview.

4.1 Staged Computation

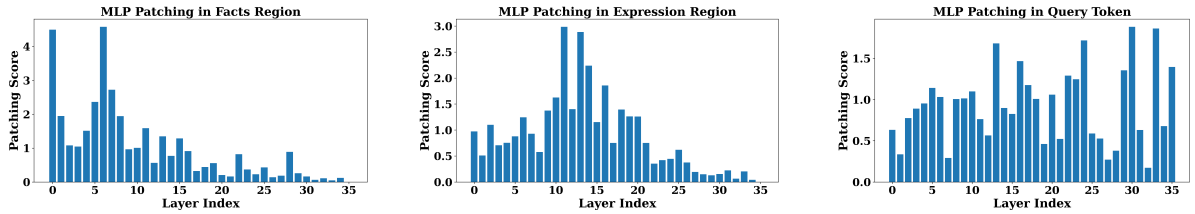
We begin with a fundamental question: *when does computation occur during logical reasoning, and what is processed at each stage?* Following prior interpretability work that identifies MLP layers as primary sites for knowledge processing (Geva et al., 2021), we employ MLP patching to reveal when different parts of the input are processed via partitioning each prompt into three semantically distinct regions: the *Facts Region* (*e.g.*, `A is True, B is False`), the *Expression Region* (*e.g.*, `(¬A or ¬B)`), and the *Query Token* (*i.e.*, `is`). For each region, we perform zero-ablation on the MLP layer outputs across all layers, quantifying the causal impact via dLD. The results are illustrated in Figure 3.

The results reveal a clear staged pattern. In early layers (L0-8), the Facts Region exhibits the highest patching effect, indicating that factual information (entity-value bindings) is primarily processed at this stage. In middle layers (L10-16), the Expression Region becomes dominant, reflecting the resolution of logical operators. The Query Token is primarily important in late layers (L24, L30, L33) where it aggregates preceding computations to form the final prediction, though it also shows notable influence in middle layers (L13). Note that precise layer boundaries vary slightly across different analytical methods (MLP patching vs. residual stream patching), but the qualitative three-stage pattern remains consistent.

This staged organization (*i.e.*, facts first, expression second, integration last) mirrors the compositional structure of the input itself. The model respects the semantic hierarchy inherent in logical statements, handling premises before conclusions. This suggests that *Qwen3* implements a systematic, compositional approach to logical reasoning, where the temporal sequence of computation aligns with the logical dependencies in the task.

4.2 Information Transmission

Tracing Information Flow. We next investigate *how information flows between computational stages* through residual stream patching, as it serves as the primary information pathway and accumu-



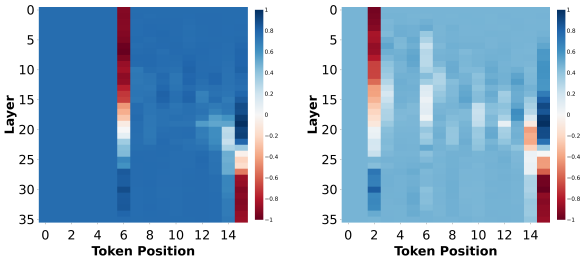
(a) Facts Region (b) Expression Region (c) Query Token

Figure 3: MLP zero-patching scores across different regions and layers on the one-hop dataset. The bar charts illustrate that the model relies on different information sources at distinct stages: (a) the **Facts Region** exhibits high sensitivity in early layers (L0-8), with patching scores declining substantially after layer 16; (b) the **Expression Region** peaks in middle layers (L10-15), reflecting logical operator processing; and (c) the **Query Token** maintains moderate but persistent influence throughout network depth, with notable peaks in both middle (L13) and late layers (L24, L30, L33), suggesting its role as the final aggregation point. More results are shown in Appendix B.1.

lates representations across layers. By patching residual stream activations at specific token positions, we can trace *where* stage-relevant information resides as it propagates through the network.

We conduct a controlled activation patching study: given a clean prompt x_{clean} (*e.g.*, `A is True, B is False, (¬A or ¬B) is`) and a corrupt prompt x_{corrupt} (*e.g.*, A is True, B is True...), we patch token-wise residual stream activations from the clean run into the corrupt run and measure the resulting shift in logit difference. This reveals which token positions carry the causal information that determines the correct answer. The resulting causal traces (Figure 4) reveal a clear pattern of information transmission. In Figure 4a, the patching effect is initially concentrated at token position 6 (*i.e.*, the truth value `True`) in early layers, then shifts to token position 14 (*i.e.*, the closing parenthesis `)`), and finally converges at the query token in late layers. Intra-segment tokens show negligible effects throughout. This trajectory indicates that information progressively aggregates at segment-terminal tokens before flowing to the final prediction position.

Pattern Quantification. We corroborate this finding with an alternative patching configuration (Figure 4b), which identifies positions 2, 6, 14, and 15 as dominant convergence points. To quantify this pattern, we analyze the mean |dLD| (*i.e.*, absolute logit difference shift) across token categories and layer groups (Figure 5). The results confirm that `facts_value` tokens dominate in early layers (L0–13), while `query_token` shows a substantial surge in late layers (L24–35), with Mean |dLD| values increasing over 2.5×. Extended experiments for *Qwen3-14B* and two-hop reasoning are



(a) B altered (True→False) (b) A altered (True→False)

Figure 4: Residual stream patching reveals information convergence. Logit difference after patching clean activations into the corrupt prompt `A is True, B is True, (¬A or ¬B) is`. Values are **normalized** per layer to highlight relative token importance. Blue regions indicate successful restoration of the correct answer. (a) Effect concentrates at token 6 (truth value), shifts to token 14 (closing parenthesis), then converges at the terminal query token. (b) Causal effects emerge at segment-terminal positions: 2, 6 (truth values), 14 (expression end), and 15 (query token).

provided in Appendix B.2 and Figure 11. These findings establish that information transmission in *Qwen3* follows a structured pattern: semantic content aggregates at the final token of each semantic segment (*e.g.*, `True` for the fact `A is True`, and `)` for the expression region) and ultimately converges at the query token (`is`). These segment-terminal tokens serve as information hubs for downstream propagation, providing the infrastructure for the staged computation observed in Section 4.1.

4.3 Fact Retrospection

An unexpected finding from the preceding analysis is that fact tokens maintain causal importance even in late layers (Figure 5 shows that `facts_value` tokens exhibit sustained comparatively high Mean |dLD| values across all layer groups). This pattern

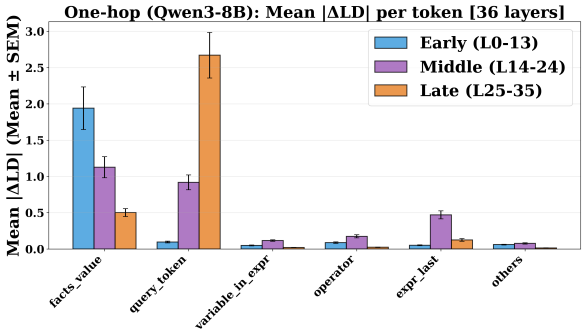


Figure 5: Token-wise information convergence in *Qwen3-8B* (One-hop). Mean |dLD| (LD shift caused by patching; see Appendix B.2) per token category across layer groups. Early layers (L0-13): `facts_value` tokens exhibit the highest causal importance, reflecting factual encoding. Late layers (L24-35): `query_token` shows a dramatic surge, indicating information convergence toward the final prediction position. Error bars denote standard error of the mean (SEM).

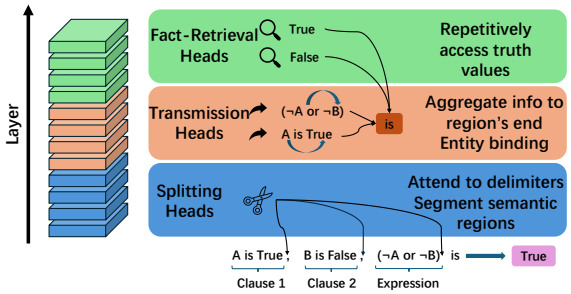


Figure 6: Schematic illustration of Specialized Attention Heads. The input sequence is segmented into semantic regions (Clauses and Logical Expression). Color coding indicates functional roles: Splitting Heads (blue) detect semantic boundaries at delimiters; Entity-Binding and Transmission Heads (orange) associate variables with values and aggregate information within regions; Fact-Retrieval Heads (green) access truth values from earlier contexts.

holds across one-hop and two-hop reasoning (Appendix B.2). We propose the Fact Retrospection hypothesis to explain this observation: rather than processing facts once and discarding the original representations, the model actively revisits source fact tokens throughout its depth. This mechanism of active information maintenance may serve several functions: (1) Error Mitigation, where retaining access to original facts allows potential correction of misinterpretations in intermediate steps; (2) Integration Facilitation, since later reasoning steps may require renewed access to earlier facts for final synthesis; and (3) Robustness Enhancement, as redundant access to source information provides resilience against representation degradation.

4.4 Specialized Attention Heads

With the macroscopic computational patterns characterized in the preceding sections, a fundamental question remains: *what specific attention heads implement these mechanisms?* We now identify such heads, and our analysis reveals precisely these patterns. We uncover Specialized Attention Heads, *i.e.*, a consistent, layer-wise division of labor in which heads at different layers assume specialized, stable functional roles that persist across logical rules and reasoning depths. We group these heads below into three main categories, each corresponding to a macroscopic mechanism identified earlier.

Splitting Heads In early layers, attention heads concentrate on comma tokens (Figure 7a), which serve as boundary markers between semantic regions (Facts vs. Expression). These heads validate Staged Computation: by detecting segment delimiters, they enable subsequent heads to process semantically coherent regions independently.

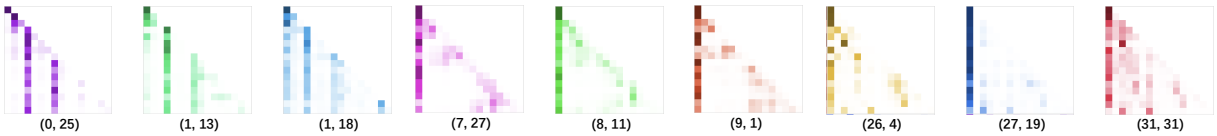
Transmission Heads These heads implement information transfer via aggregating information within semantic regions and route it toward segment-terminal tokens. Their attention patterns form lower-triangular matrices within each region (Figure 7b) and integrate toward terminal tokens. A subclass, Entity-Binding Heads, specializes in associating variables with truth values.

Fact-Retrieval Heads These heads predominantly attend to truth-value tokens (`True`/`False`) from the query token (`is`) and exhibit recurrent access patterns (Figure 7c), validating Fact Retrospection. Layer-wise analysis shows they emerge in early layers, peak in middle layers, and sustain activity in late layers (Figure 8), matching the predicted pattern of persistent fact influence.

5 Discussion

5.1 Other Attention Head Patterns.

Beyond the above three primary head types, we observe additional patterns including: (1) *Idle Heads* that consistently attend to the first token regardless of content; (2) *Information Binding Heads* in early layers (specifically, L0-10) that associate entities with truth values; (3) *Self-Processing Heads* exhibiting diagonal attention where tokens primarily attend to themselves; and (4) *Expression Processing Heads* in L18-22 that concentrate on expression tokens before attention convergence. These



(a) Splitting heads (b) Information Transmission heads (c) Fact-Retrieval heads

Figure 7: Visualization of attention matrices for three distinct types of heads. The prompt used is `A is True, B is False, (¬A or ¬B) is`. The notation (x, y) denotes the y-th head in the x-th layer. (a) The Splitting heads (early layers) attend predominantly to the comma delimiter. (b) The Information Transmission heads (intermediate layers) reveal lower-triangular structures within Fact and Expression regions. (c) The Fact-Retrieval heads (deep layers) show tokens in the Expression Region or Query Token (`is`) attending to `True`/`False` indicators.

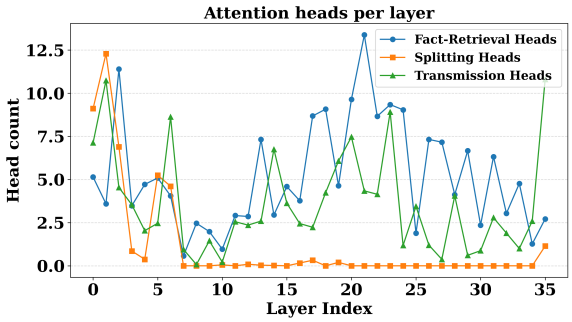


Figure 8: We set specific rules to identify the different types of attention heads, and then we calculate the number of them in our one-hop reasoning dataset. We average them in each layer. We can find that Splitting Heads mostly lie in early layers, Transmission Heads evenly lie in all the layers and Fact-Retrieval Heads mostly lie in middle and late layers. More results can be seen in Appendix B.3.

patterns, while not causally validated, suggest additional functional diversity. More Details are provided in Appendix C.1.

5.2 Functional Coupling in Specialized Heads.

We observe that individual attention heads often exhibit multiple functional roles simultaneously. For example, a head may show Splitting behavior at delimiter positions while also contributing to Transmission within semantic regions; similarly, Fact-Retrieval patterns frequently co-occur with Transmission patterns in middle-layer heads. This functional coupling suggests that our categorization of Specialized Attention Heads represents idealized abstractions rather than mutually exclusive classes. Such multi-functionality likely reflects parameter efficiency: the model implements diverse reasoning sub-processes by reusing the same heads across different computational roles. This observation aligns with the superposition hypothesis in neural networks, where individual components participate in multiple overlapping circuits. Details are provided in Appendix C.2.

6 Conclusion and Future Work

In this paper, we present a mechanistic analysis of propositional logical reasoning in *Qwen3* models, adopting a top-down analytical strategy from macroscopic patterns to microscopic implementations. Our investigation uncovers a coherent computational architecture: the model organizes reasoning into temporally distinct phases, routes information through semantic boundary tokens, maintains active re-access to source facts throughout processing, and implements these strategies through specialized attention heads. These four phenomena form an interlocking system where each mechanism supports and validates the others. Furthermore, these patterns generalize across model scales, rule types and reasoning depths, suggesting fundamental computational strategies rather than task-specific shortcuts. By characterizing *how* models reason rather than *which components* are necessary, we contribute mechanistic evidence to the hypothesis that LLMs implement algorithmic processes rather than rely solely on pattern matching.

Future Directions Our findings open several promising avenues for future research. First, extending the analysis to deeper reasoning chains (*i.e.*, three-hop and beyond) would test whether the identified mechanisms scale gracefully under increased complexity or encounter qualitative bottlenecks. Second, investigating diverse model architectures beyond the *Qwen* family (*e.g.*, Llama, Mistral, etc.) would establish the broader generality of our findings across different pre-training regimes and architectural choices. Finally, analyzing training dynamics would reveal how and when these mechanisms emerge during pretraining, *i.e.*, whether they develop gradually, appear suddenly via phase transitions (*e.g.*, grokking), or require specific training data distributions.

Limitations

Our analysis has several methodological limitations. First, while we characterize attention head specialization in detail, the internal computations within MLP layers remain largely unexplored. Our MLP patching experiments reveal when MLPs are causally important, but not what specific transformations they perform. Understanding how MLPs encode logical rules, bind variables to truth values, or compute Boolean operations represents a significant open challenge. Second, our analysis characterizes the presence of mechanisms in pre-trained models but does not explain how or when these mechanisms emerge during training. Whether these patterns arise from specific training data, emerge gradually through pretraining, or appear suddenly via phase transitions remains unknown. Finally, while we interpret our findings as evidence for structured computation, we cannot definitively rule out that these patterns reflect sophisticated statistical shortcuts that merely approximate genuine logical algorithms.

References

Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 610–623. Association for Computing Machinery.

Fengxiang Cheng, Haoxuan Li, Fenrong Liu, Robert van Rooij, Kun Zhang, and Zhouchen Lin. 2025. Empowering llms with logical reasoning: A comprehensive survey. *Preprint*, arXiv:2502.15652.

Nouha Dziri, Ximing Lu, Melanie Sclar, Xiang Lorraine Li, Liwei Jiang, Bill Yuchen Lin, Peter West, Chandra Bhagavatula, Ronan Le Bras, Jena D. Hwang, Soumya Sanyal, Sean Welleck, Xiang Ren, Allyson Ettinger, Zaid Harchaoui, and Yejin Choi. 2023. Faith and fate: Limits of transformers on compositionality. *Preprint*, arXiv:2305.18654.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, and 1 others. 2021. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 1(1):12.

Joshua Engels, Logan Riggs, and Max Tegmark. 2025. Decomposing the dark matter of sparse autoencoders. *Preprint*, arXiv:2410.14670.

Jiahai Feng and Jacob Steinhardt. 2024. How do language models bind entities in context? *Preprint*, arXiv:2310.17191.

Javier Ferrando and Elena Voita. 2024. Information flow routes: Automatically interpreting language models at scale. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 17432–17445, Miami, Florida, USA. Association for Computational Linguistics.

Mor Geva, Jasmijn Bastings, Katja Filippova, and Amir Globerson. 2023. Dissecting recall of factual associations in auto-regressive language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12216–12235, Singapore. Association for Computational Linguistics.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, and 175 others. 2025. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638.

Peter Hase, Mohit Bansal, Been Kim, and Asma Ghandeharioun. 2023. Does localization inform editing? surprising differences in causality-based localization vs. knowledge editing in language models. *Preprint*, arXiv:2301.04213.

Stefan Heimersheim and Neel Nanda. 2024. How to use and interpret activation patching. *Preprint*, arXiv:2404.15255.

Guan Zhe Hong, Nishanth Dikkala, Enming Luo, Cyrus Rashtchian, Xin Wang, and Rina Panigrahy. 2025. A implies b: Circuit analysis in llms for propositional logical reasoning. *Preprint*, arXiv:2411.04105.

Subhash Kantamneni and Max Tegmark. 2025. Language models use trigonometry to do addition. *Preprint*, arXiv:2502.00873.

Geonhee Kim, Marco Valentino, and André Freitas. 2025. Reasoning circuits in language models: A mechanistic interpretation of syllogistic inference. *Preprint*, arXiv:2408.08590.

Kenneth Li, Aspen K. Hopkins, David Bau, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. 2024. Emergent world representations: Exploring a sequence model trained on a synthetic task. *Preprint*, arXiv:2210.13382.

Samuel Marks, Can Rager, Eric J. Michaud, Yonatan Belinkov, David Bau, and Aaron Mueller. 2024. Sparse feature circuits: Discovering and editing interpretable causal graphs in language models. *ArXiv*, abs/2403.19647.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2023. Locating and editing factual associations in gpt. *Preprint*, arXiv:2202.05262.

Neel Nanda and Joseph Bloom. 2022. Transformerlens. `https://github.com/TransformerLensOrg/TransformerLens`.

Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. 2023. Progress measures for grokking via mechanistic interpretability. *Preprint*, arXiv:2301.05217.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, and 7 others. 2022. In-context learning and induction heads. *Preprint*, arXiv:2209.11895.

OpenAI, :, Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, Alex Iftimie, Alex Karpenko, Alex Tachard Passos, Alexander Neitz, Alexander Prokofiev, Alexander Wei, Allison Tam, and 244 others. 2024. Openai o1 system card. *Preprint*, arXiv:2412.16720.

Mihir Parmar, Nisarg Patel, Neeraj Varshney, Mutsumi Nakamura, Man Luo, Santosh Mashetty, Arindam Mitra, and Chitta Baral. 2024. LogicBench: Towards systematic evaluation of logical reasoning ability of large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13679–13707, Bangkok, Thailand. Association for Computational Linguistics.

Abulhair Saparov and He He. 2023. Language models are greedy reasoners: A systematic formal analysis of chain-of-thought. *ICLR*.

Lee Sharkey, Bilal Chughtai, Joshua Batson, Jack Lindsey, Jeff Wu, Lucius Bushnaq, Nicholas Goldowsky-Dill, Stefan Heimersheim, Alejandro Ortega, Joseph Bloom, Stella Biderman, Adria Garriga-Alonso, Arthur Conmy, Neel Nanda, Jessica Rumbelow, Martin Wattenberg, Nandi Schoots, Joseph Miller, Eric J. Michaud, and 10 others. 2025. Open problems in mechanistic interpretability. *Preprint*, arXiv:2501.16496.

Alessandro Stolfo, Yonatan Belinkov, and Mrinmaya Sachan. 2023. A mechanistic interpretation of arithmetic reasoning in language models using causal mediation analysis. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7035–7052, Singapore. Association for Computational Linguistics.

Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. 2021. ProofWriter: Generating implications, proofs, and abductive statements over natural language. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3621–3634, Online. Association for Computational Linguistics.

Eric Todd, Millicent L. Li, Arnab Sen Sharma, Aaron Mueller, Byron C. Wallace, and David Bau. 2024. Function vectors in large language models. *Preprint*, arXiv:2310.15213.

Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Simas Sakenis, Jason Huang, Yaron Singer, and Stuart Shieber. 2020. Causal mediation analysis for interpreting neural nlp: The case of gender bias. *Preprint*, arXiv:2004.12265.

Kevin Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. 2022. Interpretability in the wild: a circuit for indirect object identification in gpt-2 small. *Preprint*, arXiv:2211.00593.

Lean Wang, Lei Li, Damai Dai, Deli Chen, Hao Zhou, Fandong Meng, Jie Zhou, and Xu Sun. 2023. Label words are anchors: An information flow perspective for understanding in-context learning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9840–9855, Singapore. Association for Computational Linguistics.

Tingyu Xia, Bowen Yu, Yuan Wu, Yi Chang, and Chang Zhou. 2024. Language models can evaluate themselves via probability discrepancy. *Preprint*, arXiv:2405.10516.

Jiaran Ye, Zijun Yao, Zhidian Huang, Liangming Pan, Jinxin Liu, Yushi Bai, Amy Xin, Weichuan Liu, Xiaoyin Che, Lei Hou, and Juanzi Li. 2025. How do transformers learn implicit reasoning? *Preprint*, arXiv:2505.23653.

Zeping Yu and Sophia Ananiadou. 2024. Interpreting arithmetic mechanism in large language models through comparative neuron analysis. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 3293–3306, Miami, Florida, USA. Association for Computational Linguistics.

Fred Zhang and Neel Nanda. 2024. Towards best practices of activation patching in language models: Metrics and methods. *Preprint*, arXiv:2309.16042.

Honghua Zhang, Liunian Harold Li, Tao Meng, Kai-Wei Chang, and Guy Van den Broeck. 2022. On the paradox of learning to reason from data. *Preprint*, arXiv:2205.11502.

Contents of Appendix

A Experimental Setup Details

A.1 Hardware and Software

Hardware. All experiments are conducted on NVIDIA H100 GPUs with 96GB memory. The large memory capacity allows us to process the entire *Qwen3-14B* model in float16 precision without requiring model parallelism.

Software Environment. We use PyTorch 2.0 with CUDA 11.8. Activation patching experiments are implemented using the TransformerLens library (Nanda and Bloom, 2022), which provides convenient hooks for accessing and modifying intermediate activations in transformer models.

A.2 Implementation Details

Activation Patching Procedure. For each patching experiment, we perform three forward passes:

1. **Clean run**: Execute the model on x_{clean} (correct prompt) and cache activations at all layers and positions.

2. **Corrupted run**: Execute the model on $x_{corrupt}$ (modified prompt with altered truth values) to obtain baseline logits.

3. **Patched run**: Re-execute $x_{corrupt}$, but replace specific activations with cached values from the clean run at targeted components (residual stream, attention heads, or MLP outputs), then measure the resulting shift in logit difference.

Patching Granularities. We perform patching at three levels of granularity:

- **Residual Stream**: Patch resid_pre$_l^i$ at layer l and position i before the layer's computation.

- **Attention Heads**: Patch the output $z_{l,h}$ of head h at layer l for all token positions.

- **MLP Outputs**: Patch mlp_out$_l^i$ at layer l and position i after the MLP computation.

All forward passes are performed with `torch.no_grad()` to reduce memory consumption and disable gradient computation. We use `float16` precision for all activations and model weights.

Computational Cost. A single residual stream patching experiment (36 layers $\times$ approximately 16 token positions) takes approximately 30 seconds on an H100 GPU for *Qwen3-8B*. Attention head patching (36 layers $\times$ 32 heads) requires approximately 2 minutes. The total computation time for all experiments presented in this paper is approximately 50 GPU-hours.

A.3 Reproducibility

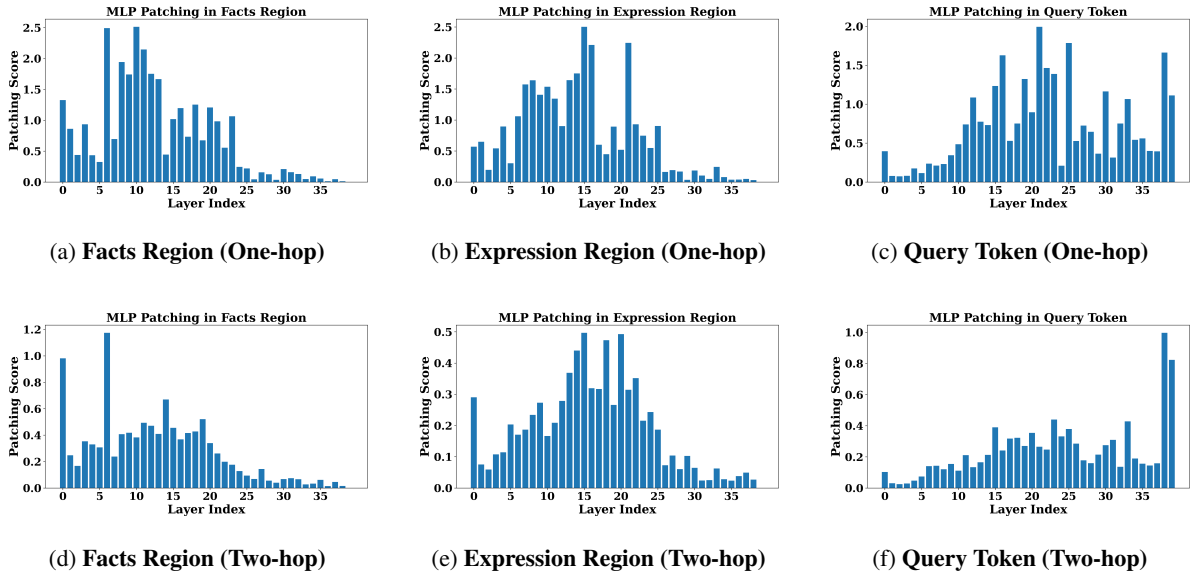
Code and data will be made available upon publication. Our implementation is based on publicly available models from Hugging Face and the open-source TransformerLens library, ensuring that all experiments can be reproduced by the community.

A.4 Dataset Details

PropLogic-MI consists of paired clean/corrupted prompts for activation patching experiments. Each sample includes:

One-Hop Reasoning. Prompts follow the format: "`A is [True/False], [expression] is`", where the expression directly evaluates the logical rule. For example, identity law: "`A is True, A and True is`" (expected: True). Clean/corrupted pairs differ by flipping one variable's truth value, ensuring minimal perturbation for causal analysis.

Two-Hop Reasoning. Prompts include an intermediate variable: "`A is [T/F], B is [expr of A], C is [T/F], [expr of B,C] is`". For example: "`A is True, B is A and True, C is False, B and C is`" requires first computing $B = A \wedge \top =$ True, then $B \wedge C =$ False.



(a) **Facts Region (One-hop)** (b) **Expression Region (One-hop)** (c) **Query Token (One-hop)**

(d) **Facts Region (Two-hop)** (e) **Expression Region (Two-hop)** (f) **Query Token (Two-hop)**

Figure 9: Logit difference shift ratio (R_{LD}) via MLP zero-ablation in *Qwen3-14B*. **Top Row (a-c): One-hop dataset.** A staged pattern emerges: (a) the **Facts Region** is key in early layers, (b) the **Expression Region** dominates middle layers, and (c) the **Query Token** prevails in middle-to-late layers. **Bottom Row (d-f): Two-hop dataset.** We observe a phenomenon consistent with the one-hop results, confirming the robustness of the observed pattern.

Corruption Strategy. For each sample, we generate a corrupted version by flipping exactly one fact variable's truth value. This controlled perturbation enables precise causal attribution: when patching restores the correct answer, we can attribute the effect to the specific flipped variable.

Filtering Criteria. We retain only samples where the model correctly predicts both clean and corrupted prompts. This ensures that activation patching measures genuine mechanism disruption rather than pre-existing model failures.

B More Results

B.1 Staged Computation

We replicated our MLP zero-ablation experiments on *Qwen3-14B* to verify cross-model consistency. Following the same methodology as Section 4.1, we partitioned prompts into Facts Region, Expression Region, and Query Token, then measured the logit difference shift ratio:

$$R_{\text{LD}} = |(\text{LD}_{\text{after}} - \text{LD}_{\text{origin}})/\text{LD}_{\text{origin}}|$$

where $\text{LD}_{\text{origin}}$ and LD_{after} denote the logit difference before and after zero-ablation. Results (Figure 9) confirm that the staged computation pattern generalizes across model scales.

B.2 Information Transmission

Metric Definitions. We define the **logit difference shift** (dLD) as the change in logit difference caused by activation patching:

$$\text{dLD}_{l,i} = \text{LD}^{(l,i)}_{\text{patched}} - \text{LD}_{\text{baseline}}$$

where $\text{LD}^{(l,i)}_{\text{patched}}$ denotes the logit difference after patching the clean activation at layer l and position i into the corrupted forward pass. We employ two aggregation strategies:

(1) Mean |dLD| across samples (Figure 5 and Figure 11): We aggregate |dLD| by: (i) averaging over layers within each stage (Early, Middle, Late), (ii) averaging over tokens belonging to the same category within each sample, and (iii) averaging across all samples in the dataset.

(2) Mean |dLD| per token (M|dLD|, Figure 10): For single-sample case studies, we compute the mean absolute shift at each token position i within a layer stage S:

$$\text{M}|\text{dLD}|^{(S)}_i = \frac{1}{|S|}\sum_{l \in S} |\text{dLD}_{l,i}|$$

This metric captures how much patching each token position shifts the model's prediction within each layer group.

Single-Sample Case Studies. Figure 10 extends the single-sample analysis from Figure 4 (main

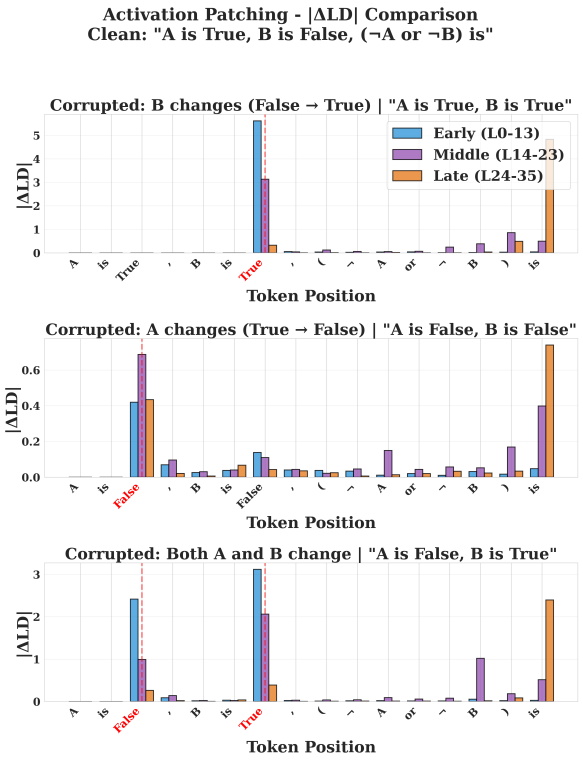


Figure 10: Token-wise activation patching under multiple corruption scenarios. Each panel shows M|dLD| per token across Early (L0-13), Middle (L14-23), and Late (L24-35) layer groups. The clean prompt is `A is True, B is False, (¬A or ¬B) is`. Top: B changes (False→True). The corrupted `True` token (position 8) exhibits high sensitivity in early/middle layers. Middle: A changes (True→False). The corrupted `False` token shows similar early-layer dominance. Bottom: Both A and B change. Combined effects appear at both fact positions. Across all scenarios, the query token `is` shows a significant surge in late layers, confirming information convergence toward the prediction position.

text) by systematically varying which fact tokens are corrupted. This provides a controlled validation of the Information Transmission phenomenon across different corruption configurations.

The results reveal several consistent patterns across all corruption scenarios:

(1) Early-layer fact encoding. In each scenario, the corrupted fact token(s) exhibit maximal patching sensitivity in early layers (L0-13). When only B changes (Top panel), position 8 dominates; when only A changes (Middle panel), position 4 dominates; when both change (Bottom panel), both positions show elevated effects. This confirms that fact tokens are primarily encoded in early layers regardless of which specific facts are manipulated.

(2) Late-layer query convergence. Across all three scenarios, the query token `is` consistently shows a pronounced surge in late layers (L24-35), with M|dLD| values of +2.0, +0.8, and +2.5 respectively. This invariance across corruption types provides strong evidence that late-layer query convergence is a robust architectural property, not an artifact of specific input configurations.

(3) Additive effects in multi-fact corruption. The Bottom panel (both A and B corrupted) shows that patching effects at individual fact positions are approximately additive: the combined effect at positions 4 and 8 roughly equals the sum of effects observed when each is corrupted individually. This suggests that the model processes multiple facts through parallel, largely independent early-layer pathways before integrating them at later stages.

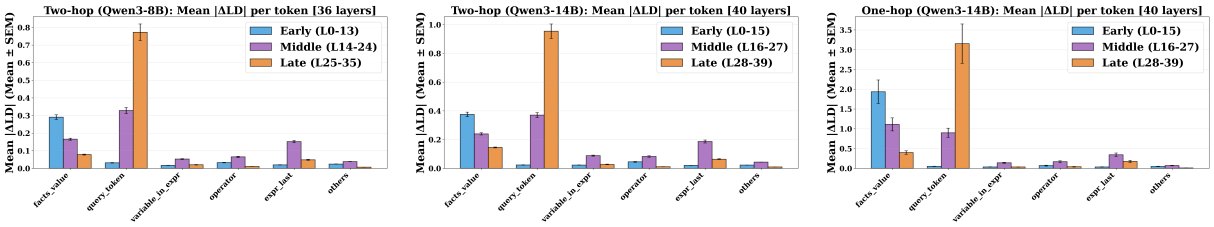
Extended Results. To verify the generalizability of the Information Transmission phenomenon, we extend our token-wise |dLD| analysis to *Qwen3-14B* and two-hop reasoning tasks. Figure 11 presents the complete results.

Cross-Model Consistency. Comparing Figure 5 (*Qwen3-8B*) with Figure 11c (*Qwen3-14B*), both models exhibit nearly identical convergence patterns on one-hop tasks. The `facts_value` dominance in early layers and `query_token` surge in late layers are preserved across the 8B→14B scaling, suggesting that Information Transmission is an architectural property rather than a scale-dependent phenomenon.

Reasoning Depth Effects. In two-hop reasoning (Figures 11a and 11b), the overall |dLD| magnitudes are substantially lower compared to one-hop tasks. This attenuation reflects the increased computational complexity: with additional intermediate reasoning steps, causal effects become more distributed across the network. Nevertheless, the qualitative convergence pattern, characterized by early factual encoding followed by late-layer query aggregation, remains intact. Furthermore, the `expr_last` token (representing the final token of the expression region) exhibits moderate but consistent |dLD| values in middle layers across all settings, reinforcing our finding that segment-terminal tokens serve as critical information hubs throughout the network.

B.3 Specialized Attention Heads Distribution

To verify cross-scale generalization, we applied the same attention head classification to *Qwen3-14B*.



(a) Two-hop (*Qwen3-8B*) (b) Two-hop (*Qwen3-14B*) (c) One-hop (*Qwen3-14B*)

Figure 11: Token-wise information convergence across model scales and reasoning depths. All panels show Mean |dLD| per token category across Early (L0-13), Middle (L14-23), and Late (L24-35) layer groups. (a-b) Two-hop reasoning: The convergence pattern persists with attenuated magnitudes due to increased reasoning complexity. Notably, `query_token` remains dominant in late layers, while `expr_last` (the segment-terminal token of the expression region) also shows elevated |dLD| in middle layers, confirming that segment-terminal tokens serve as information hubs. (c) Qwen3-14B One-hop: Consistent with the 8B model (Figure 5), demonstrating scale-invariant information flow patterns. These results confirm that Information Transmission is a robust mechanism across both model scales and reasoning depths.

Figure 12 presents the layer-wise distributions for both one-hop and two-hop reasoning tasks.

The results demonstrate strong consistency with *Qwen3-8B* (Figure 8): Splitting Heads remain concentrated in early layers (L0-15), Transmission Heads distribute across middle layers (L10-30), and Fact-Retrieval Heads peak in middle-to-late layers (L15-40). This parallel structure across model scales provides strong evidence that the Specialized Attention Heads phenomenon reflects a fundamental organizational principle of transformer-based logical reasoning, rather than an idiosyncratic property of a specific model checkpoint.

Notably, the Fact-Retrieval Heads distribution in *Qwen3-14B* further validates the Fact Retrospection hypothesis: these heads maintain sustained activity into late layers, consistent with the persistent causal influence of fact tokens observed in our residual stream analysis. The results are shown in Figure 12.

B.4 Failure Case Analysis

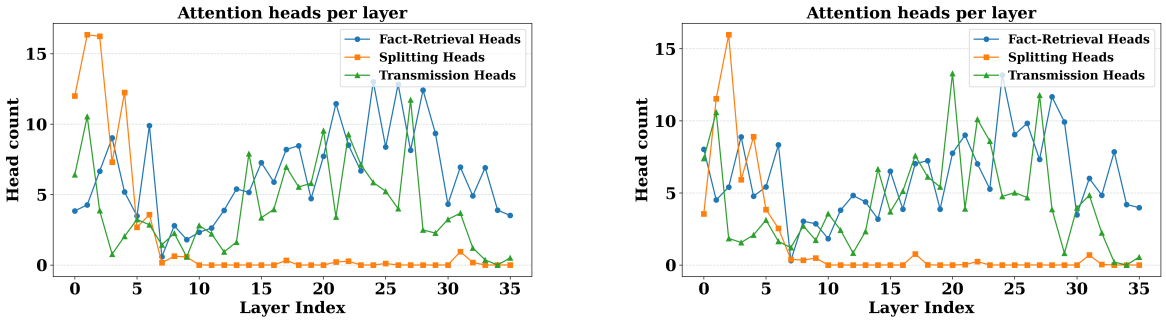
We analyze cases where the model produces incorrect outputs despite exhibiting the same functional head profiles observed in successful reasoning. This analysis reveals that failures are not due to a breakdown in head specialization, but rather to limitations in downstream integration.

Persistent Head Specialization. Even when the model fails on logical reasoning tasks involving excessive reasoning depth or element overload, attention heads continue to exhibit their characteristic functional profiles: Splitting Heads still concentrate on delimiters, Transmission Heads still aggregate within regions, and Fact-Retrieval Heads still attend to truth values. This persistence suggests that the identified mechanisms represent robust architectural properties rather than task-contingent behaviors.

Potential Failure Modes. We hypothesize three potential sources of failure:

- **Interaction Deficiency**: Insufficient interaction between heads with complementary roles, preventing effective information handoff.
- **Integration Misalignment**: Misalignment between head outputs and subsequent MLP computations, leading to incorrect logical combinations.
- **Dynamic Rigidity**: The model's inability to dynamically reconfigure attention patterns when faced with novel or compositional reasoning demands.

Chain-of-Thought Recovery. Notably, in certain failure cases, allowing the model to generate a sufficiently long CoT enables it to ultimately arrive at the correct answer. This suggests that some failures in implicit (non-CoT) reasoning stem from premature convergence or incomplete information aggregation at the query position. Extended sequential processing may compensate by providing additional computational steps for information integration. This observation invites further investigation into the relationship between reasoning depth, head collaboration, and the emergence of correct solutions.



(a) **One-hop Dataset** (b) **Two-hop Dataset**

Figure 12: Layer-wise distribution of functional attention heads in *Qwen3-14B* across different reasoning tasks. Panels (a) and (b) illustrate the distributions on the one-hop and two-hop datasets, respectively. The distributions mirror those observed in *Qwen3-8B* (Figure 8): **Splitting Heads** concentrate in early layers, **Transmission Heads** span middle layers, and **Fact-Retrieval Heads** peak in middle-to-late layers. This cross-scale consistency confirms that Specialized Attention Heads represent an architectural property rather than a model-specific artifact.

C Discussion

C.1 Other Attention Head Patterns

Beyond the three primary head types (*i.e.*, Splitting, Transmission, and Fact-Retrieval) identified in our main analysis, we observe additional attention patterns that provide a more complete picture of the model's internal organization.

Idle Heads. As network depth increases, certain heads exhibit minimal functional contribution, consistently attending to the first token regardless of input content. Figure 13(a) shows Layer 7 Head 4, where attention concentrates heavily on the first column—all tokens attend primarily to position 0. To verify this is not an artifact of specific token semantics, we tested prompts where the first token is a semantically neutral period (".") rather than a meaningful character. The pattern persists. Furthermore, we observed that L7H4 exhibits identical idle behavior in multi-hop reasoning tasks and even in unrelated prompts (e.g., "The weather is sunny, I am"), suggesting that idle heads may be an intrinsic property of the model architecture rather than task-specific behavior. This raises two possibilities: (1) certain heads are permanently idle across all inputs, representing genuinely redundant capacity; or (2) some heads may be conditionally idle, remaining inactive for certain task types while activating for others. A comprehensive analysis of all idle heads across diverse prompts would be needed to distinguish these hypotheses.

Information Binding Heads. In early layers (L0–10), we observe heads that associate entities with their truth values. Specifically, these heads bind "A" with "True" and "B" with "False" in the Facts region, and bind negation operators ("¬") with their operands in the Expression region. Figure 13(b-c) illustrates two examples from Layer 0: Head 1 and Head 24 both exhibit diagonal patterns with localized binding, where adjacent tokens within semantic units show elevated mutual attention. Since tokenization separates these semantically coupled elements (e.g., "¬" and "A" are distinct tokens), such binding appears necessary for downstream logical operations. This binding mechanism complements the Fact-Retrieval heads identified in middle-to-late layers.

Self-Processing Heads. Some heads exhibit strong diagonal attention patterns, where each token primarily attends to itself with minimal attention to other positions. Figure 13(d-e) shows two examples: Layer 12 Head 20 and Layer 22 Head 12. Both display sharp diagonal lines with near-zero off-diagonal attention, indicating that each token's representation is refined independently. This self-focused processing may facilitate residual-like refinement of token representations without cross-position information mixing. Such heads appear across various layers, suggesting a distributed mechanism for token-level representation enhancement.

Expression Processing Heads. In layers 18–22, immediately preceding the attention convergence observed in our main analysis, we identify heads that concentrate attention within the expression region tokens. Figure 13(f-h) presents three examples

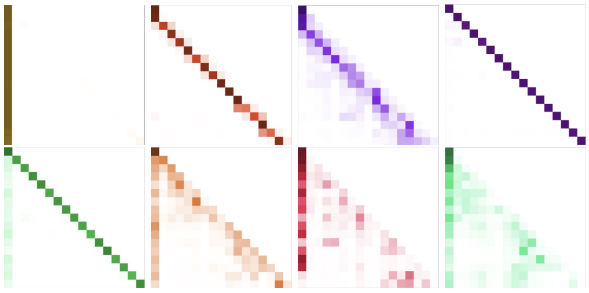


Figure 13: Attention patterns of other head types in *Qwen3-8B*. Top row: (a) Idle Head L7H4—attention concentrates on position 0; (b-c) Information Binding Heads L0H1, L0H24—diagonal patterns with local binding; (d) Self-Processing Head L12H20—sharp diagonal. Bottom row: (e) Self-Processing Head L22H12; (f-h) Expression Processing Heads L18H2, L19H31, L21H13—attention concentrated in expression region (lower-right).

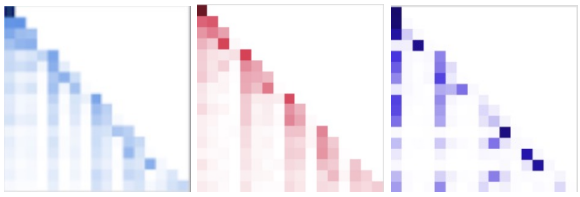


Figure 14: Examples of functional coupling in early-layer heads. Left: L0H19 exhibits staircase-like attention blocks indicating both splitting (segment boundaries) and information binding (within-segment attention). Middle: L0H31 shows diagonal self-processing combined with localized binding blocks. Right: L1H22 displays diagonal attention with scattered cross-position connections, suggesting simultaneous self-processing and selective information retrieval. All three heads demonstrate that individual attention heads participate in multiple functional roles.

from Layers 18, 19, and 21. These heads show elevated attention in the lower-right region of the attention matrix, corresponding to the expression tokens ("(¬A or ¬B) is"). Notably, Layer 19 Head 31 additionally attends to the first token, exhibiting functional coupling between expression processing and positional anchoring. These heads appear to process the logical structure of expressions, potentially executing core logical computations. The layer positioning aligns with our Staged Computation findings: expression processing occurs after fact encoding (early layers) but before final answer aggregation (late layers).

Relationship to Primary Mechanisms. These additional patterns complement rather than contradict our main findings. Idle Heads may explain why not all attention capacity is utilized for reasoning. Information Binding Heads provide the early-layer foundation that enables later Fact-Retrieval. Self-Processing Heads may contribute to the residual stream's role in information maintenance. Expression Processing Heads bridge the gap between Information Transmission and final prediction, aligning with the Staged Computation framework. While these patterns have not been causally validated through patching experiments, they suggest functional diversity beyond the three primary head types and warrant further investigation.

C.2 Functional Coupling in Specialized Heads

We provide visual evidence for the functional coupling phenomenon described in the main text. Figure 14 shows three examples from early layers, each exhibiting multiple functional characteristics simultaneously.

These attention patterns reveal that early-layer heads commonly exhibit multi-functional profiles:

- **L0H19**: Staircase-like attention blocks at segment boundaries combined with elevated within-segment attention, performing both splitting and information binding.
- **L0H31**: Diagonal emphasis (self-processing) with localized attention blocks within "A is True" and "B is False" segments (information binding).
- **L1H22**: Strong diagonal pattern with scattered high-attention points at specific cross-positions, suggesting self-processing combined with selective fact retrieval.

This multi-functional profile supports the superposition hypothesis: individual heads participate in multiple overlapping computational circuits, achieving parameter efficiency through functional reuse.