

Structure from Sequence: Mechanisms of Tabular Retrieval

Siddharth Parekh

CMU-CS-26-123

August 2026

Computer Science Department
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:

Carolyn Rosé, Chair
Daphne Ippolito

*Submitted in partial fulfillment of the requirements
for the degree of the Masters of Science in Computer Science.*

Keywords: mechanistic interpretability, table question answering, transformer circuits, activation patching, faithfulness, large language models

For Hemali, Parag and Hriday

Abstract

Structured objects such as tables, JSON records, and tool outputs reach decoder-only language models as flat token sequences. This creates a *structural addressing* problem: locating the relevant part before using its content. I study single-cell table lookup as a controlled case through three linked questions: which cues make a row-column address recoverable and which errors they repair; whether lookup decomposes into identifiable internal operations; and whether causal interventions support that decomposition and distinguish it from simpler accounts of format effects.

I evaluate real tables in a 48-condition grid spanning two open-weight families, table formats, prompts, and row-index conditions, combining behavioral evaluation, attention analysis, and causal interventions. Explicit numeric row indices improve exact-match accuracy by 16–33 percentage points in every tested condition, chiefly by repairing wrong-row errors. Reversal and aligned-pair label swaps indicate that they can serve as content-addressable rather than ordinal landmarks. Across both families, attention profiles place column-name matching before correct-cell retrieval; ablation and activation patching in the primary Llama regime provide causal support for this two-stage account. The division of labor recurs across families, but head identities and causal concentration differ: the named heads form a causal core, not a complete circuit. Format interventions further implicate exact, nameable content rather than pipe counting in the localized column-matching route.

Although the evidence covers only short, flat tables with numeric row keys in an atomic lookup task, it suggests that representations can act as computational scaffolds. Exposing stable landmarks and testing how models use them could inform structured-data interfaces designed jointly for accuracy, robustness, and inference cost.

Acknowledgments

First and foremost, I would like to thank my advisor, Professor Carolyn Rosé, for her unwavering support and guidance — not only throughout this work, but over the past three years. My summer internship with her gave me my first real experience of AI research and, despite the twists and turns along the way, helped me discover the career I hope to pursue. Through several projects with Professor Rosé and the Teledia Lab, particularly in collaboration with Dr. Armineh Nourbakhsh, I found both my intellectual niche and a community in which I could grow. They have been the most generous mentors a once-lost college student could have asked for.

I am also grateful to Professor Daphne Ippolito for serving on my committee and for offering thoughtful insights into the broader implications of this work, as well as the many interesting directions in which it might be extended.

I am deeply grateful to Zhen Wu and Dr. Ritam Dutt for their time, effort, and generosity, and for giving me the best collaborative research experience I could have hoped for. My conversations with them continually challenged me to articulate my ideas more clearly and to examine the intuitions underlying my research. Even amid the demands of their full-time work, they continued to make time to advise and support me throughout the summer. Their commitment to both this project and my growth as a researcher has meant more to me than I can adequately express.

I am indebted to my friends and family for standing by me through what I can only describe as an incredibly difficult year. Their patience, encouragement, and care made it possible for me to keep moving forward.

Finally, I would like to thank the broader Carnegie Mellon community. This place has made me the person I am today, and it would be incomplete without everyone—from its professors and students to its support staff and administrative workers—playing their part. These five years have been profoundly transformative and as I reach the end of this chapter, all I can say is that my heart is in the work.

Contents

1	Introduction	1
1.1	Motivation: Structural Addressing from Sequence	1
1.2	The Cell-Lookup Task as a Minimal Case	3
1.3	Thesis Statement and Contributions	4
1.4	Scope and Organization	5
2	Background and Related Work	7
2.1	Table Question Answering and Structure-Aware Models	7
2.2	Large Language Models over Serialized Tables	8
2.3	Mechanistic Interpretability: Methods and Circuits	9
2.4	Positioning: Mechanistic Table Reading and the Delimiter-Counting Account . .	10
3	Task, Data, and Methods	13
3.1	The Structural Question-Answering Task	13
3.2	Tables, Serialization, and Row-Index Conditions	14
3.3	Models	15
3.4	Evaluation and Analysis Methods	16
3.4.1	Behavioral Scoring and Error Classification	16
3.4.2	Attention-Based Measures	17
3.4.3	Causal Interventions	18
3.4.4	Attribution and Circuit Evaluation	20
3.4.5	Candidate-Head Selection	21
3.5	Reproducibility	22
4	Behavioral Results	23
4.1	Accuracy Across Formats and Prompts	23
4.2	Effects of Explicit and Reversed Row Indices	23
4.3	Stability to Index-Column Position	27
4.4	Failure Modes: Off-by-Row and Column Geometry	28
4.5	Base versus Instruct	29
5	Observational Evidence for Matching and Retrieval	31
5.1	A Two-Stage Depth Profile	31
5.2	Within-Table Column Alignment	32

5.3	Question-to-Table Routing	33
5.4	Attention to the Target Cell	34
5.5	Relating Attention to Behavior	35
5.6	Attention in Wide Tables	36
6	Causal Tests of the Candidate Circuit	39
6.1	Mean-Ablation Necessity	39
6.2	Activation Patching and Marginal Repair	41
6.3	Joint Ablation and Direct-Logit Writers	43
6.4	Scalable Node Attribution and Circuit Ranking	45
6.5	Faithfulness and Circuit Adequacy	45
6.5.1	Faithfulness Against Circuit Size	46
6.5.2	Inverse-Complement Necessity	49
6.5.3	Tier-Level Minimality	49
6.5.4	Dependence on Metric and Ablation Flavor	50
7	Alternative Explanations for Format and Model Differences	51
7.1	The Pipe-Deletion Control	51
7.2	Lexical versus Semantic Matching	52
7.3	Tokenization Controls	53
7.4	A Training-Data Hypothesis	54
8	Formatting Guidance	57
8.1	Recommendations the Mechanism Supports	57
8.2	Provisional Recommendations	58
8.3	Limits of Formatting Interventions	59
9	Conclusion and Future Work	61
9.1	Summary	61
9.2	Interpretation and Scope	62
9.3	Limitations	63
9.3.1	External Validity	63
9.3.2	Sample and Metric Provenance	63
9.3.3	Uncertainty and Controls	64
9.4	Future Work	64
A	Supplementary Behavioral Results	67
A.1	Reversed Row Order on the Original Example Bank	67
A.2	Marginal Column-Position Analyses	68
B	Observational Maps and Head Rankings	69
B.1	Retrieval-Pool Maps	69
B.2	Query-Token Routing Maps	71
B.3	Cross-Family Replication Maps	71
B.4	Tuning Stability and Co-Localization	73

B.5	Top-Ten Rankings for the Primary Metrics	76
C	Circuit Robustness and Measurement Details	79
C.1	Faithfulness Grids	79
C.2	Answer-Token Audit	81
	Bibliography	83

List of Figures

1.1	The same table in three representations.	2
4.1	Exact-match accuracy across the condition grid.	24
4.2	The row-index intervention.	25
4.3	Error composition with and without a row index.	26
4.4	Stability to index-column position.	27
4.5	Column geometry by width, position, and header class.	29
4.6	Three local contrasts: prompt packaging, shots, and weights.	30
5.1	Column matching precedes retrieval across 48 conditions.	32
5.2	Attention alignment to column and row structure.	33
5.3	Question-token routing to table landmarks.	34
5.4	The late, diffuse retrieval stage.	35
5.5	Attention strength versus behavior across conditions.	36
6.1	Mean-ablation necessity by stage.	40
6.2	Two error families by retrieval attention.	41
6.3	Activation-patching rescue by site and corruption axis.	42
6.4	Joint necessity and the writer null.	44
6.5	No single MLP is a large marginal bottleneck.	44
6.6	Node-level attribution ranking.	45
6.7	Interchange-faithfulness curves.	47
6.8	Cross-family necessity and sufficiency.	48
6.9	Tier-level minimality: dropping a tier from the evaluation circuit.	50
7.1	QK selection margins survive pipe deletion.	52
7.2	The named QK route does not survive near-synonym column names.	54
A.1	Row anchoring under reversed row order, original bank.	67
A.2	Marginal accuracy by column position.	68
B.1	Structural composition of final-token attention.	70
B.2	Full per-head routing maps with lexical controls.	71
B.3	Qwen Instruct roster-defining maps.	72
B.4	Base-Instruct rank stability by metric.	73
B.5	Metric co-localization in Llama.	74

B.6 Metric co-localization in Qwen.	75
---	----

List of Tables

3.1	The experimental condition grid.	15
3.2	Aligned-pair yield by corruption axis.	20
3.3	Tested candidate-head rosters with selection scores.	22
4.1	Two-shot prompting under reversed row order.	27
5.1	Attention geometry in the pooled widest-table bin.	38
7.1	Tokenizer statistics across families.	54
A.1	The complete demonstration-bank grid.	68
B.1	Top ten heads by attention alignment.	76
B.2	Top ten heads by query-token routing.	77
B.3	Top ten heads by retrieval metrics.	78
C.1	Llama faithfulness grid.	80
C.2	Qwen faithfulness grid.	80
C.3	Axis-resolved faithfulness.	81
C.4	Answer-token audit counts.	81

Chapter 1

Introduction

A decoder-only language model receives a serialized table as a flat token sequence, but answering a cell-lookup question requires recovering both a row and a column. This thesis asks how models recover and use that address. I organize the study around three linked questions: which input cues make the address recoverable and which errors they repair; whether lookup decomposes into identifiable internal operations; and whether causal interventions support that decomposition and distinguish simpler explanations for format effects.

Together, the questions define an evidence chain: behavioral contrasts identify useful cues, attention analyses trace candidate routes, and interventions test whether those routes matter. Within the tested single-cell task, the evidence supports a two-stage functional organization: *matching* binds the queried column name and, in indexed conditions, the requested numeric row key to landmarks in the serialized table; *retrieval* uses the resulting address to select the target cell.

1.1 Motivation: Structural Addressing from Sequence

The difficulty is that the row–column geometry a person reads off a table at a glance is not explicitly marked in the model’s input. A dataframe exposes coordinates directly: a cell is addressed by a row key and a column key, and the surrounding software guarantees that the addressing is exact. A serialized table hands the model only a flat run of tokens in which structural roles must be inferred from indirect cues: the delimiters between cells, the newline tokens between rows, the header tokens in the first line, and the fact that cells in one column recur at a roughly fixed stride through the sequence. None of these cues is labeled as such, so recovering a usable address from them is a genuine in-context computation, not a coordinate the tokenizer already provides.

Figure 1.1 shows the gap. The object a person pictures (a) is a two-dimensional grid; what the model receives (b) is a single string in which row and column boundaries survive only as newline and pipe characters; and what actually enters the network (c) is a sequence of integer token identifiers, in which those delimiters may stand alone or be merged with adjacent whitespace and newline characters into larger subword units. Any row–column address the model uses to answer a question about the table must be inferred from representation (c).

Tables are a clean setting in which to study structural addressing. They are simple enough to support a precisely scorable task—a cell-lookup question has one target cell, fully determined by

(a) As a two-dimensional grid, the object a reader pictures.

Row	City	Score
1	Boston	82
2	Chicago	91
3	Seattle	77

(b) As the one-dimensional string the model actually receives. The only real line breaks are the `\n` markers; the wrapping here is for display.

```
| Row | City      | Score |\n| 1    | Boston   | 82    |\n
| 2    | Chicago  | 91     |\n| 3    | Seattle  | 77     |\n\n
Question: What is the value in the 'Score' column, row 2?\nAnswer:
```

(c) After tokenization, as integer token identifiers (61 in all, first 20 shown), with the string each identifier decodes to underneath.

```
[91, 11035, 765, 4409, 262, 765, 18607, 9432, 91, 220, 16, 256,
 765, 10406, 220, 765, 220, 6086, 262, 9432, ...]
```

```
'|' ' Row' ' |' ' City' ' ' ' |' ' Score' ' ' |\n' ' |' ' ' ' 1'
' ' ' |' ' Boston' ' ' ' |' ' ' ' 82' ' ' ' |\n' ...
```

Figure 1.1: The same table in three representations. A reader pictures the two-dimensional grid (a); the model receives the one-dimensional string (b), in which row and column boundaries survive only as newline and pipe characters; and the network operates on the integer token identifiers (c). Panel (b) is the exact serialization this thesis evaluates, including the fixed question template (Section 3.1); the Instruct runs wrap it in the model’s chat template. Panel (c) is the Llama-3.1-8B tokenization of that string. It shows why a row–column address cannot be read directly from token boundaries: the same pipe character is sometimes its own token (91, at the start of a line) and sometimes merged with the preceding space and the following newline into a single unit (9432 = ‘ |\n’), so no token boundary reliably marks a cell boundary.

the prompt—yet structured enough that the answer cannot be determined from the question alone. Answering *the value in the Score column, row 2* requires jointly locating the requested row and column in the supplied table and returning the value at their intersection. That combination of an unambiguous target and a joint row–column dependency is what makes tables a useful minimal case for the broader question of how sequence models extract latent structure from flat text.

The same problem recurs, in richer form, across much of what language models are asked to do. A JSON object is a flat sequence with latent hierarchy; source code is a flat sequence with scope, binding, and control flow; a game transcript is a flat sequence with a latent board state. In each case the model must recover and use structure that the surface form only implies. Whether sequence models build such latent objects at all is the question behind work on emergent world representations, where a model trained only on game moves is found to encode board state internally [19]. Tables pose a humbler version: does a decoder trained on ordinary text

recover and use a row–column address when all it observes is a serialized grid? The operation it would need is reminiscent of induction—locate a landmark in context, bind a query token to it, and use that binding to drive a later prediction [29]—but applied to two interacting axes rather than a single repeated pattern. Locating candidate attention heads for this operation, testing their causal roles, and then asking how adequately the resulting circuit description captures a potentially distributed computation is the methodology this thesis adopts, following the indirect object identification circuit in GPT-2 [39].

The question is not only of theoretical interest. Deployed models routinely read structured data pasted directly into a prompt (CSVs, markdown tables, spreadsheet fragments, tool outputs, log lines), and their native ability to infer structure governs how robust they are when that data is reformatted. Behavioral studies have repeatedly found this ability to be brittle and format-dependent: accuracy on the same underlying table shifts substantially with serialization format, content order, and delimiter choice [22, 35].

A mechanism-level account could turn such surface choices into testable representation decisions: identify the tokens that function as structural landmarks, verify the route by which a model uses them, and evaluate the representation jointly for accuracy, robustness, and inference cost. This points toward mechanism-aware interfaces for model-facing structured data. The present thesis takes a controlled first step by studying lookup accuracy and causal mechanism; adaptive interfaces and their token, latency, and cost effects remain future work.

1.2 The Cell-Lookup Task as a Minimal Case

The task studied throughout is deliberately plain. The model is given a table serialized as text and a question whose answer is the value of a single cell (for instance, *What is the value in the Score column, row 2?* for the table of Figure 1.1), and is scored by exact string match against that cell’s contents. The answer is determined entirely by the prompt: no world knowledge, arithmetic, or multi-hop reasoning is required or rewarded. This makes successful structural processing directly scorable, although a correct output alone does not establish how it was produced. The evidence is strongest when the target value is unique within its table, so that no other cell could have produced it by chance; this holds for most sampled cells, and it is the subset on which Chapter 4 re-checks the central result. The tables are real Wikipedia and WTQ tables that may appear in pretraining, and no nonce-table or content-permutation control was run, so recall of a memorized table cannot be excluded by the task design alone. The causal evidence adds sensitivity to the mapping inside the prompt: on the selected aligned-pair subset, swapping row-index labels usually makes the model follow the relabeled row (Section 6.2). This is harder to explain by a fixed memorized association, although it does not exclude pretraining overlap. Because the target is a single cell rather than an aggregate or a paraphrase, the internal computation can be tied to concrete operations: identifying a column, identifying a row, and retrieving their intersection.

The empirical study holds the lookup task fixed while varying table format, row-index condition, prompt, and model family. Tables are drawn from a corpus of real tables, rather than synthetic grids, and serialized in two formats, markdown and CSV. Two model families are examined, Llama-3.1-8B [23] and Qwen-2.5-7B [34], each in its base and instruction-tuned variants,

all run through TransformerLens [28] so that attention patterns and residual-stream activations are directly inspectable. The empirical analysis follows the three questions in sequence. Behavioral exact-match and error analyses first identify which cues help and which failures they repair across the 48-condition grid. Attention alignment, routing, and layer-depth profiles then locate candidate matching and retrieval operations. Finally, ablation, activation patching, and circuit-adequacy analysis test their causal roles; delimiter deletion and a question-side near-synonym substitution test competing explanations for the format effects. Chapter 3 specifies the corpus, conditions, models, and methods in full.

Single-cell lookup over a short, flat table is intentionally the atomic case, isolated so that additional operations such as arithmetic or multi-hop reasoning cannot explain success. The thesis does not aim to cover the entirety of table understanding: aggregation, multi-hop joins, and hierarchical or very long tables lie beyond it, and Chapter 2 situates the task within that larger landscape. Section 1.4 states what it *is* the atomic case *of*.

1.3 Thesis Statement and Contributions

Within the tested single-cell setting, the evidence supports a **two-stage account of structural addressing**. An earlier *matching* stage relates the queried column name—and, in indexed conditions, the requested numeric row index—to landmarks in the serialized table. A later *retrieval* stage uses the resulting address to select the target cell and move its value toward the output. Attention profiles show this functional ordering across both model families, although causal evidence is deepest in the primary Llama regime. The named heads form a causal core within a broader distributed implementation; what recurs across families is the division of labor, not head coordinates or the concentration of causal responsibility.

The thesis makes four main contributions.

A behavioral account of structural lookup. An explicit numeric row-index column substantially improves lookup across the condition grid, chiefly by repairing wrong-row errors. Position, reversal, and label-swap tests show that the tested indices can act as content-addressable landmarks rather than merely restating rendered row order. Markdown is another robust behavioral advantage in the tested Llama grid, while the width and target-position results remain observational (Chapter 4).

A mechanistic decomposition. Across all 48 completed conditions, attention analyses place column-name matching before correct-cell retrieval; in indexed conditions, a separate route binds the requested row index to its table occurrence. The same functional division recurs across Llama and Qwen without requiring the same head coordinates (Chapter 5).

Causal tests of the proposed circuit. In the primary Llama intervention regime, ablation and activation patching support distinct matching and retrieval roles. Cross-family joint-ablation and heads-only interchange tests show why necessity and sufficiency must be reported separately: the named heads form a causally important core, but an adequate heads-only account is much more distributed. The two model families distribute that responsibility differently (Chapter 6).

A test of competing explanations and practical implications. Pipe deletion shows that the localized column-matching route does not primarily count markdown delimiters, although removing them is not behaviorally free. A near-synonym instead weakens the named lexical route. These results support row-index columns and markdown within the tested task; recommendations about other key types, table geometry, and header classes remain provisional (Chapters 7 and 8).

1.4 Scope and Organization

Single-cell lookup is the atomic case of flat-table reading: every query that reads a value must first address it, so an account of addressing is a component of any account of the harder queries built on top of it. That is the sense in which the results here are meant to generalize, and it is a claim about the operation rather than about the implementation. What the evidence bounds is correspondingly narrower. The measurements cover two open-weight families at seven to eight billion parameters, real but short and flat Wikipedia tables (median 11 rows by 6 columns), and row keys that are numeric indices in forward or reversed order. Behavioral and observational results span the full 48-condition grid, whereas the main circuit-adequacy interventions use one primary regime and detailed activation patching is restricted to Llama. Mean ablation scores generated accuracy; patching, attribution, and faithfulness score an outcome-conditioned first-token logit margin. Section 9.3 collects these constraints, and Section 9.2 returns to which findings should be expected to travel.

The remainder proceeds as follows. Chapter 2 places the work against table question answering and mechanistic interpretability. Chapter 3 specifies the corpus, conditions, models, and methods. Chapter 4 reports the behavioral grid, Chapter 5 the attention analyses that locate the two stages, and Chapter 6 the interventions that test them. Chapter 7 tests the delimiter-counting alternative and examines the cross-family difference, Chapter 8 separates the mechanism-backed formatting guidance from the provisional, and Chapter 9 returns to the three questions, bounds the resulting claims, and develops a path from structural landmarks toward mechanism-aware interfaces for reliable and efficient structured reasoning.

Chapter 2

Background and Related Work

This thesis sits at the intersection of two research directions that have so far developed largely in parallel: the study of how language models process tabular data, and the mechanistic interpretability of transformer circuits. The first tells us *what* models do with tables and *when* they fail; the second supplies the vocabulary and the causal methodology for asking *how* they do it internally. The object of study here is intentionally narrow: how an ordinary decoder-only model reconstructs row-column geometry from a serialized table and routes a lookup question to the right cell. Answering even that narrow question well, however, requires inheriting standards from both literatures. This chapter reviews table question answering and structure-aware modeling (Section 2.1), the LLM-era study of serialized tables as prompts (Section 2.2), and the mechanistic-interpretability methods and circuits that this work builds on (Section 2.3). Section 2.4 states the position of this thesis relative to its nearest work, the concurrent mechanistic analysis of cell location by Zhang et al. [44].

2.1 Table Question Answering and Structure-Aware Models

Question answering over tables predates the current generation of large language models and initially grew out of semantic parsing and table-aware representation learning. Pasupat and Liang [30] framed compositional question answering over semi-structured Wikipedia tables (WikiTableQuestions) as a benchmark for reasoning under weak supervision. Iyyer et al. [14] recast the problem as sequential, conversational QA (SQA) that requires resolving contextual dependencies across a question sequence, and Zhong et al. [45] scaled supervision dramatically by pairing natural-language questions with executable SQL over a large collection of relational tables (WikiSQL). Later datasets broadened the task family beyond value lookup: TabFact targets fact verification against tables [3], HybridQA requires multi-hop reasoning that joins a table to linked free text [4], and FeTaQA elicits free-form answers grounded in supporting cells [27]. Together these define the core operations—row selection, column selection, aggregation, and compositional reasoning—against which any account of table understanding is measured.

A second line of work improved performance by injecting table structure directly into model architecture or pretraining. TaBERT jointly pretrains over textual context and tabular content in a shared encoder [41]; TAPAS extends BERT with table-aware input handling and weakly

supervised cell selection plus aggregation, dispensing with explicit logical forms [10]; TURL introduces a structure-aware transformer with masked entity recovery [6]; TUTA uses tree-based attention for generally structured, spreadsheet-like tables [40]; TABBIE emphasizes table-only pretraining rather than text–table pairing [13]; TAPEX casts pretraining as learning a neural SQL executor [21]; and OmniTab mixes natural and synthetic data to improve few-shot and fully supervised table QA [16].

These models are essential background, but they are also methodological foils for the present work. Each makes table structure *more explicit to the model* than plain-text prompting ordinarily does, whether through table-native input encodings, row/column-aware attention, or structure-oriented pretraining objectives. This thesis asks the complementary question: given only a plain-text serialization and no table-specific architecture, how does a standard decoder-only LLM recover enough latent structure to answer a lookup query at all? The contribution is not a better table encoder, a new pretraining recipe, or a richer input representation, but a mechanistic account of unaided reading.

2.2 Large Language Models over Serialized Tables

With instruction-following LLMs, the question shifted from specialized encoders to how much table understanding emerges from general-purpose models prompted on serialized tables. Chen [2] showed that large language models are surprisingly strong few-shot table reasoners, especially with chain-of-thought prompting, moving attention from table-specific architectures toward native in-context reasoning over linearized tables.

Subsequent work studied this capability systematically, and its central finding is precisely the phenomenon this thesis seeks to explain: the serialization format is not a neutral wrapper around the same information. Sui et al. [35] built a structural-understanding benchmark and showed that performance depends strongly on input format, content order, role prompting, and partition marks. Liu et al. [22] sharpened this by testing structural perturbations and combining textual and symbolic reasoning, finding that structure normalization matters and that aggregating reasoning paths substantially improves results. These are behavioral, black-box regularities; the present work can be read as a mechanistic attempt to explain why they arise internally. The behavioral chapter of this thesis reproduces the same family of effects on real tables (markdown is consistently easier than CSV across all matched conditions, and an explicit row-index column is the strongest single formatting lever) and then traces each effect to specific internal operations rather than leaving it at the level of prompt sensitivity.

A parallel line treats tables less as text to be read and more as objects to be queried through interfaces, programs, or tools. StructGPT lets an LLM read structured data iteratively through specialized interfaces [15]; Cao et al. [1] translate table questions into Python over multi-index Pandas data frames with optional APIs; and TableLLM trains a dedicated model for office-style spreadsheet and document manipulation [43]. These approaches often improve robustness on heterogeneous or complex tables, but they do so by *externalizing* structure into tools, code, or specialized weights, factoring out exactly the native reading mechanism this thesis aims to characterize.

Recent benchmarks also mark the boundary of the present claims. LongTableBench probes

long-context, multi-format, multi-domain table reasoning [20], and StructFact documents persistent weaknesses in factual reasoning over heterogeneous structured data [11]. These results position a mechanistic account of short, flat, single-cell lookup as a necessary first step rather than a complete theory of table understanding: the atomic structural core, isolated so that success reflects structure recovery rather than local next-token statistics, with transfer to wider and more hierarchical settings left as future work.

2.3 Mechanistic Interpretability: Methods and Circuits

The immediate mechanistic background comes from the transformer-circuits tradition. Elhage et al. [7] formalized compositional analysis of attention heads, and Olsson et al. [29] characterized induction heads as a reusable sequence operation: locate an earlier occurrence of a pattern, then copy or continue it. Table lookup is not induction, but it has a similar flavor: the model must find a structural landmark in context, bind a query token to it, and use that binding downstream. Wang et al. [39] then reverse-engineered a full natural-language behavior, indirect object identification (IOI) in GPT-2 small, into a sparse circuit of named attention heads evaluated for faithfulness, completeness, and minimality. This thesis follows the same ambition on a structured-data task: find the heads, characterize their roles, intervene on them, and hold the resulting circuit to adapted IOI-style adequacy tests.

A related thread asks whether sequence models build latent world models. In Othello-GPT, Li et al. [19] show that a model trained only on move sequences develops an internal representation of board state. Tables raise a more modest version of the same question: does a decoder trained on text build internal row and column coordinates when all it observes is a serialized table? The two-stage matching-then-retrieval pipeline documented here is one concrete answer: the model constructs and then consumes structural bindings that are nowhere given explicitly in the input.

Methodology matters as much as the circuits themselves, because an attention map or decodable representation can show that information is present without showing that the model uses it. An early intervention-based NLP study made that distinction explicit through causal mediation analysis: Vig et al. [38] treated neurons and attention heads as mediators of gender-bias behavior. Meng et al. [24] then developed causal tracing for factual recall, corrupting an input and restoring selected clean hidden states to locate where the answer could be recovered. The IOI analysis extended the same logic from sites to an end-to-end language circuit, combining mean ablation, activation and path patching, and direct-logit decomposition to assign roles and test the resulting head set [39]. These precedents motivate three different questions in this thesis: ablation asks whether a component is needed, denoising patching asks whether its clean state can repair a particular corrupt computation, and a circuit-wide intervention asks whether a selected set can collectively reproduce the measured behavior. None is treated as a generic “importance” score.

The replacement distribution and output metric define each of those questions. Zero ablation removes a component with a fixed value; mean ablation replaces its prompt-specific contribution with a dataset reference; resample or interchange ablation supplies an activation realized on another input. Likewise, an intervention can be scored by loss, probability, a contrastive logit difference, or generated accuracy. Systematic comparisons show that activation-patching local-

izations can change with the corruption, patched site, and metric [42], while circuit faithfulness can change sharply with the ablation flavor [25]. The replacements and metrics this thesis selects among those options, and the caveats they carry, are specified once in Section 3.4.

Scaling causal localization motivated automated and gradient-based alternatives. Exhaustively patching every head, MLP, position, or edge requires a separate intervention for each candidate. ACDC automates circuit discovery by recursively pruning edges with repeated patching tests [5]; it produces an explicit subgraph, but its repeated edge interventions and threshold choices are poorly matched to an 8B model with roughly thousand-token prompts. AtP* approximates component patching with gradients while addressing common false negatives [18]. Edge attribution patching uses a first-order approximation to rank circuit edges with only a small number of forward and backward passes [37]; EAP-IG applies the integrated- gradients construction [36] along the corrupt-to-clean path to improve the faithfulness of the selected circuit [9]. This thesis adapts that principle to whole attention heads and MLP modules (Section 3.4); the gradient score is a scalable screen, not a substitute for an intervention.

Finally, circuit discovery requires evaluating a set rather than merely recovering familiar nodes. Wang et al. [39] framed that evaluation in terms of faithfulness, completeness, and minimality, and Hanna et al. [9] argued that preservation of behavior should take precedence over overlap with a previously proposed circuit. Interchange interventions have a separate lineage in causal abstraction, where internal states realized on a source input are inserted into a base input to test whether a representation plays the role of an interpretable causal variable [8]. MIB standardizes faithfulness-versus-size evaluation and related circuit metrics [26]. This thesis follows that curve-based protocol on its own task, using example-conditioned interchange because it supplies a constructed all-head endpoint and an empirically checkable zero-head endpoint, and reports the ablation flavor and metric on every faithfulness result. Its inverse-circuit and tier-drop experiments are inverse-complement necessity and partial minimality tests, respectively; they are evidence relevant to circuit adequacy, not proofs of completeness or of the stronger component-wise criteria.

2.4 Positioning: Mechanistic Table Reading and the Delimiter-Counting Account

The closest contemporary work is Zhang et al. [44], a concurrent mechanistic analysis of cell location that also studies Llama-3.1-8B on cell-lookup questions. Methodologically, the studies share more than the task: both use head ablation, semantic row/column counterfactuals, and activation patching scored by a normalized clean-versus-corrupt logit difference. Their analysis uses synthetic tables, zero ablation, residual-stream and head-output patches, linear probes, and steering vectors. This thesis instead uses real tables and scores mean ablation on full generation; its patching adds cell-content and row-index counterfactuals, exact token-alignment checks, the denoising direction, and separate query, attention-pattern, and head-output interventions, then evaluates ranked circuits at multiple sizes. These differences matter because the intervention literature above predicts that corruption, site, and metric can change the localization.

Substantively, Zhang et al. [44] propose a three-stage account (semantic binding, coordinate

localization, and information extraction) in which column localization is implemented by *counting pipe delimiters*. Where they explore delimiter counting as the operative mechanism, this thesis tests that account directly and finds content-based matching on the localized route.

Relative to the broader landscape, then, prior table-QA work largely optimized performance by adding architectural bias, pretraining objectives, or tool use; prior LLM table studies documented prompt and format sensitivity without explaining its internal basis; and prior mechanistic work established that sequence models can represent latent structure, only very recently targeting tables directly. This thesis connects these strands with a circuit-level account of how generic decoder-only LLMs recover table geometry from plain-text serialization, holding that account to MIB-comparable faithfulness standards and testing it against the nearest competing mechanistic hypothesis.

Chapter 3

Task, Data, and Methods

This chapter defines the structural question-answering task, the tables and serialization formats used, the models under study, and the evaluation and analysis methods that the observational and causal chapters draw on. The question template, question sampling, decoding, and scoring rule are held fixed across all conditions; the manipulated factors are model family, tuning and prompt packaging, serialization format, shot count, and row-index condition.

3.1 The Structural Question-Answering Task

The unit of study is a single *cell-lookup* question. The model is shown a table serialized as text, followed by a question that names a column and a row and asks for the value at their intersection (for the table of Figure 1.1, *What is the value in the `Score` column, row 2?*), and its output is scored by exact string match against that cell’s contents. Questions are generated programmatically from the table so that the target is always a genuine cell, and each table contributes up to fifteen questions covering different row/column coordinates.

Prompts, decoding, and scoring. The question is generated from a fixed template, *What is the value in the ‘ $\langle \text{column name} \rangle$ ’ column, row $\langle N \rangle$?*, and questions are sampled uniformly without replacement from the table’s valid (row, column) cells with a seeded generator; when a row-index column is present it is never itself a question target. Chat-style prompts wrap the table and question in the model’s own chat template behind a system instruction to respond with only the cell value; base-style prompts prepend a one-line instruction and, in the few-shot conditions, two fixed worked examples rendered under the same format and row-index condition as the main table. Decoding is greedy (temperature 0), capped at 30 new tokens; the answer is the first generated line, truncated at any generated “Question:”. This truncation isolates the model’s response to the current question, since cell values are single-line and greedy decoding often continues past the answer to imitate the prompt’s format with a self-generated “Question:”.

Exact match is case-insensitive equality of the whitespace-stripped answer and cell string; every accuracy in this thesis is this exact-match figure.

The answer is determined by the prompt: the question names a column and a row, the target is a specific cell of the serialized table, and the task requires no world knowledge, arithmetic,

or multi-hop inference. When the target value is unique in the table (which holds for 57.7% of sampled cells), an exact match identifies the queried coordinate. For the rest (a 3.5% empty-cell fraction and a 38.8% fraction whose answer string also appears elsewhere in the table) an exact match need not mean the right coordinate was found; these cases are separated out as a robustness check in Section 4.2, where the row-index result only strengthens. Because the target is a single cell, success decomposes into concrete, localizable operations—find the column axis, find the row axis, return their intersection—that the later chapters tie to specific internal computations. Exact-match scoring is strict; the small fraction of errors in which the model retrieves the right value but renders it in a different surface form is tracked as its own failure category (Chapter 4), not counted as correct.

3.2 Tables, Serialization, and Row-Index Conditions

The tables are drawn from the WikiTableQuestions corpus of Wikipedia tables [30], so any mechanism this thesis identifies must survive the irregularity of naturally occurring headers, column widths, and cell contents. From 1,384 tables, the 44 with duplicate column headers are excluded (a header-named question would be ambiguous), leaving 1,340; prompts that would exceed a model’s context window are additionally skipped at evaluation time. The full 1,340-table set is used for evaluation with no train/test division: the models are never fine-tuned on these tables. The retained tables have a median of 11 data rows (interquartile range 8–15, maximum 34) and 6 columns (IQR 5–7, maximum 25), and 30% of cells are numeric; answers are therefore a mix of numbers, names, and short phrases, all scored as strings. Each table is serialized in two formats: **markdown**, with pipe-delimited cells, and **CSV**, with comma-delimited cells and no markdown-style header-separator row. The two formats carry identical information but differ in their delimiter tokens and column alignment, which turns out to matter behaviorally (Chapter 4) and mechanistically (Chapter 7).

The single most important manipulation is the **row-index condition**, which controls whether and where the serialized table carries an explicit row-number column. Three conditions are used throughout:

- **none**—the table has no row-index column; rows are distinguished only by their content and their position in the sequence.
- **rowid**—an explicit row-index column (1, 2, 3, . . .) is prepended as the first column.
- **randidx**—the same row-index column is inserted at a *randomized* column position rather than first, detaching the index’s *content* from its *location*.

A supplementary fourth condition, **revidx** (markdown only, both families), renders the data rows in reverse so that the row-index column reads $N, \dots, 1$ top-to-bottom while “row N ” is still scored against the row *labeled* N . This pits index content against physical order and is used in Chapter 4 to separate row counting from row matching. Its two-shot cells use a ten-row example bank rather than the three-row bank of the main grid, so that both demonstrations have targets whose rendered position moves under reversal; the reversed zero-shot cell for Qwen base was not run. Two recorded covariates describe table geometry: **table width** (the number of data columns) and **queried-column position**. They are not randomized controls. Chapter 4 conditions on them

Factor	Levels		
Family	Llama-3.1-8B	Qwen-2.5-7B	
Tuning/prompt package	base weights + text	Instruct weights + chat	
Format	markdown	CSV	
Row-index	none	rowid	randidx (+ revidx suppl.)
Shots	0-shot	2-shot	

Table 3.1: The experimental condition grid. The 48 cells are 2 families $\times$ 2 tuning/prompt packages $\times$ 2 table formats $\times$ 3 row-index conditions $\times$ 2 shot counts; prompt style is not a sixth independent factor. Primary circuit-adequacy interventions use Instruct, markdown, row-index, chat zero-shot; a matched-prompt diagnostic and a Llama base two-shot ablation check are reported separately.

to separate a marginal “far-right column” pattern into a broad width association and an additional late-position penalty in wider tables. Each behavioral grid cell aggregates 1,340 tables $\times$ up to 15 questions ($n \approx 20,100$ questions).

The numeric index is a controlled proxy for a natural row key: it is unique, present in every row, and independent of the queried data column. Moving between **none** and **rowid** changes the available addressing information from rendered order alone to an explicit key match. Row lookup keyed by a name or date is not tested. Section 9.2 discusses how these conditions relate to naturally keyed tables.

3.3 Models

Two model families are studied, each in its base and instruction-tuned variants: **Llama-3.1-8B** [23] and **Qwen-2.5-7B** [34]. Studying two families is essential to the central cross-family claim of Chapter 6: a mechanism found in only one model could be idiosyncratic, whereas a division of labor that recurs across two independently trained families, at different layer/head coordinates, is harder to attribute to one network’s particulars. All models are run through **TransformerLens** [28], which exposes attention patterns and residual-stream activations at every layer and head and supports the activation caching, patching, and ablation hooks that the causal chapter relies on. Llama-3.1-8B has 32 layers $\times$ 32 heads (1,024 attention heads); Qwen-2.5-7B has 28 $\times$ 28 (784 heads). Head indices are written $LxHy$ for layer x , head y , zero-indexed.

The main 48-cell grid varies shots (zero-shot or two-shot), but does not vary prompt style independently of tuning: base weights use the base-style text prompt, while Instruct weights use the model’s chat template. In the Instruct two-shot condition, examples are separate user/assistant turns. A separate matched-prompt diagnostic runs Instruct weights with the base-style text prompt to isolate the weight change from prompt packaging. Table 3.1 summarizes the resulting condition grid. Holding these factors explicit is what lets Chapter 4 attribute the instruction-tuned model’s apparent advantage to prompt format rather than to tuned weights.

3.4 Evaluation and Analysis Methods

Each method is introduced here and developed in the chapter that uses it. The analyses run on different samples, score different outcomes, and carry different controls and notions of uncertainty; where a result depends on one of these choices, the dependence is stated alongside the result.

3.4.1 Behavioral Scoring and Error Classification

Full accuracy grids across family, tuning, format, shots, and row-index condition (Chapter 4) establish the phenomena to be explained, together with an error taxonomy that later chapters map onto specific mechanisms. Because one of the central claims is stated in terms of that taxonomy, its buckets are defined operationally and applied as an ordered cascade in which the first match wins, so every answer lands in exactly one bucket. Writing g for the model’s answer and t for the target cell, both case-folded, punctuation-stripped, and whitespace-collapsed, and (R, C) for the target coordinate in the table as the model saw it (row-index column re-inserted at the position that condition used):

1. **correct**—the answer is an exact match under the scoring rule above.
2. **empty**— g is the empty string.
3. **format_mismatch**— $g = t$ after normalization: the right value in the wrong surface form. This precedes every table-geometry bucket, so a correct value is never counted as a mislocation.
4. **table_mismatch**—the archived table does not reconstruct the recorded target $((R, C)$ out of range, or the cell there is not t). A bookkeeping bucket; it flags examples the taxonomy cannot adjudicate, not model failures.
5. **off_by_row**—some *other* data row $r \neq R$ has g in column C : the right column, the wrong row. The signed offset $r - R$ of the nearest such row is recorded. The bucket counts any wrong row, not only adjacent ones.
6. **wrong_col_same_row**—some other column $c \neq C$ has g in row R : the right row, the wrong column.
7. **present_elsewhere**— g occurs somewhere in the table but in neither the target row nor the target column.
8. **not_in_table**— g occurs nowhere in the table.

The precedence matters for interpretation: because `off_by_row` precedes `wrong_col_same_row`, an answer that is simultaneously in another row of the target column and another column of the target row is counted as a row error. The same classification procedure is used for the behavioral breakdowns and the attention-by-failure analyses, so the taxonomy is consistent between chapters.

The column-geometry analysis is a post-hoc reanalysis of the archived per-question records from 33 **none/rowid** conditions: the 32 cells obtained from model variant, format, shot count, and those two row-index conditions, plus Llama Instruct zero-shot with a space-aligned **rowid** table. The stored column index and width describe the rendered, augmented table, so one is

subtracted from both in **rowid** conditions to recover data-column coordinates. **randidx** is excluded because its inserted index position varies by table, making the corresponding rendered data-column index unstable. The main position comparison contrasts data columns 0–2 with the last three data columns at each exact width; at widths four and five those bands overlap (sharing two columns and one, respectively), which attenuates the contrast by construction, so the position comparison is read from width six upward. The binned per-index curves are estimated within each condition by table-cluster bootstrap and summarized by their median across conditions; a curve point requires at least 20 underlying tables in that condition. The exact-width comparison instead pools all conditions, requires at least 300 condition-question records on each side, and bootstraps by underlying table-file stem, thereby keeping repeated formats and conditions for a source table in one cluster. Separate robustness summaries compute within each configuration (i) the early-column change from widths 8–10 to ≥ 11 and (ii) the columns-0–2 versus columns-9+ contrast inside the ≥ 11 bin, then report the median change and its sign count. For a secondary descriptive split, a table is labeled **numeric/empty** when at least half of its header cells are blank or numeric-like, and **lexical** otherwise. Width, position, and header class are all observed properties, not interventions, so their contrasts are associations and remain open to column-semantic and source-template confounding. The 33 configurations reuse the same underlying tables, so counts such as 32/33 describe directional stability across settings, not independent replications; clustering by table stem also does not model higher-level source-template dependence.

3.4.2 Attention-Based Measures

Attention alignment. For each head, the attention mass a query position places on tokens of its own row (or column) is compared with the mass expected under uniform attention over the causally visible prefix: for query position q and a row mask m , the score is $\sum_{i \leq q} a_{qi} m_i - \frac{1}{q+1} \sum_{i \leq q} m_i$, averaged over query positions and tables. Heads whose attention tracks the table’s row and column geometry far above this causal-uniform baseline (Chapter 5) become the candidate matching and retrieval heads.

Question-token routing. Routing measures which question tokens bind to which table landmarks (Chapter 5). Both sides of the measurement are token *spans*, since column names, row numbers, headers, and cells are all frequently multi-token, and each side is reduced in a different way. On the question side, four spans are located by character offset and mapped to every token whose character range overlaps them: the column name (the text inside the quotes of the question template), the literal word “column”, the literal word “row”, and the digit run naming the row. Anchoring on the quotes first and searching the keywords forward from there prevents a column name that happens to contain the word “row” from being confused with the keyword. On the table side, a landmark (a header cell, an index cell, or the answer cell) is the set of tokens overlapping that cell’s characters. The score for one (query span, landmark) pair is the attention mass *summed* over the landmark’s key tokens and then *averaged* over the query span’s tokens, so a multi-token column name contributes the mean routing of its pieces rather than their sum, and scores stay comparable across names of different token lengths. Scores are then averaged over examples. This is what shows that rows are anchored by index content rather than by name.

The retrieval-side metrics are the final token’s attention mass on the correct cell and its *structural precision*, the correct-cell mass as a fraction of the mass on all structurally related cells (the correct cell plus its row and column).

3.4.3 Causal Interventions

Three intervention families support the causal analysis (Chapter 6). Mean ablation asks whether a component is needed for full generation; clean-to-corrupt patching asks whether restoring its state can repair a particular counterfactual; and circuit-wide interchange asks whether a selected set can reconstruct that counterfactual when its complement is corrupted. Gradient and direct-logit attribution rank or nominate components for those tests. This division follows the circuit-analysis program exemplified by Wang et al. [39]; the sensitivity of patching and faithfulness to the corruption, replacement distribution, and metric [25, 42] is treated as part of the experimental design. Unless stated otherwise, an intervention on a head is applied at its output (z).

Mean-ablation. Mean ablation has been used to test the behavioral necessity of proposed transformer circuits [39]. I use it here for full-generation necessity because it can be applied across the unpaired task evaluation and preserves the error taxonomy. Zero ablation would remove the head contribution entirely but need not resemble a task activation, while counterfactual resampling would require a paired donor and ask a pair-specific question. The chosen reference is the head’s dataset mean, computed per (layer, head, dimension) over all token positions of 64 sampled task prompts. It removes prompt-specific variation while retaining a task-average contribution, but is still a synthetic point that need not be produced by any prompt. The result is necessity under this replacement, not context-free indispensability. The intervention runs during a 200-table $\times$ 8-question generation evaluation over the first 200 lexicographically sorted corpus files, not a randomized table sample. It is compared with a random-head control drawn equal to or larger than the ablated set (twelve random heads for the four-head tiers in both families; sixteen for Llama’s twelve named heads and fifteen for Qwen’s, seed 7). One draw is used, not several, and it is not layer-matched, so it is a sanity check rather than a calibrated null (Section 9.3.3).

Aligned counterfactual pairs. Activation patching and the later attribution/faithfulness analyses use the same pair-construction rule but *different saved samples*. The patching run constructs up to 300 pairs per axis directly with a 1300-token sequence cap. Node attribution uses a separately saved 300-pair-per-axis bank capped at 1024 tokens, and each faithfulness run selects 150 pairs from every axis of that bank. A clean prompt and a minimally corrupted twin are accepted only if they tokenize to identical length with identical token ids outside the edited span, so that a patched activation at position i addresses the same content in both runs. Four corruption axes are used: (a) the column name in the question, (b) the row number in the question, (c) the target cell’s content, and (d) the row-index values of the target and a donor row. For pair i , let y_i^+ and y_i^- be the first tokens of the clean and corrupt answers, required to be distinct. Because BPE vocabularies tokenize `answer` and `answer` differently, the implementation tokenizes both variants and takes the first token of whichever the model assigns the higher logit. This rule conditions on the measured logits, so it can bias pair acceptance and every margin built on it in the

optimistic direction. A post hoc audit against the deterministic token (read from the tokenization of `prompt + answer`) agreed with the chosen token in 4,734 of 4,738 recoverable cases across both banks; Section 9.3.2 records the audit and the residual caveat.

$$M_{r,i} = \text{logit}_{r,i}(y_i^+) - \text{logit}_{r,i}(y_i^-).$$

A pair is retained only if $M_{\text{clean},i} > 0 > M_{\text{corrupt},i}$: the model prefers the clean answer in the clean world and the alternative answer in the corrupt world. In the saved attribution/faithfulness banks, the pooled mean margins are +14.26/ − 13.69 for Llama and +23.17/ − 21.86 for Qwen.

For activation patching, denoising rescue and reverse-noising damage are

$$r_i = \frac{M_{\text{patched},i} - M_{\text{corrupt},i}}{M_{\text{clean},i} - M_{\text{corrupt},i}}, \quad d_i = \frac{M_{\text{clean},i} - M_{\text{noised},i}}{M_{\text{clean},i} - M_{\text{corrupt},i}}.$$

The normalization is performed separately for each pair and the resulting fractions are then averaged. Patching and faithfulness intervals bootstrap the pooled per-pair array rather than resampling source tables, so they treat pairs as independent and can be too narrow where one table contributes several pairs. The mean-ablation and joint-ablation effects of Chapter 6 carry no interval of either kind: that evaluation recorded aggregate accuracy and failure composition rather than per-question records, so those accuracy deltas are point estimates. A rescue of 0, 1, or more than 1 means no movement from the corrupt endpoint, recovery of the full clean–corrupt gap, or overshoot beyond the clean endpoint, respectively; values are not clipped. Rescue is therefore a fraction of a selected-pair first-token logit gap, not the fraction of questions corrected.

That conditioning selects a subpopulation. Table 3.2 gives the attempted denominator and the acceptance rate for every axis and both models. Roughly half of all attempted questions survive (55% in Llama, 45% in Qwen), but the rate varies from 76% on the row-number axis to 23% on the cell-content axis, and the reasons differ by axis. The content axis loses most of its candidates before any model runs: many cells have no same-length replacement elsewhere in the table, and the substitution more often breaks token alignment. The row-number and row-index axes lose theirs mostly to the requirement that the answer and the corrupt answer differ in their *first* token, which a digit-per-token tokenizer violates far more often; this is the largest single reason Qwen’s yield is lower than Llama’s throughout. Only a small remainder is lost to the confidence conditions themselves ($\leq 5\%$ per axis), with one informative exception reported as a result in Section 6.2: on the row-index-swap axis, 18 of 318 otherwise-valid Llama pairs fail the $M_{\text{corrupt}} < 0$ condition because the model keeps preferring the clean answer after the swap: it follows the original physical position rather than the relabelled row. The remaining $\approx 94\%$ follow the label. The first-token results of Chapter 6 therefore apply to pairs meeting the token-alignment, distinct-first-token, and margin-reversal criteria. Whether the same head assignment governs the excluded cases (long multi-token answers, cells with no alignable substitute) is not measured here.

Activation patching. Causal tracing and circuit analyses have used activation patching to localize where a clean internal state can repair a corrupted behavior [24, 39]. Here it runs over the separately constructed 1300-token-cap pairs described above. Clean activations are written into the corrupt run to measure *marginal counterfactual sufficiency*; the reverse, corrupt-into-clean

corruption axis	Llama-3.1-8B-Instruct		Qwen-2.5-7B-Instruct	
	attempted	accepted	attempted	accepted
(a) column name in question	413	300 (73%)	437	300 (69%)
(b) row number in question	394	300 (76%)	450	300 (67%)
(c) target-cell content	952	300 (32%)	1289	300 (23%)
(d) row-index value swap	411	300 (73%)	498	300 (60%)
all axes	2170	1200 (55%)	2674	1200 (45%)

Table 3.2: Aligned-pair yield for the saved 1024-token-cap attribution/faithfulness banks. “Attempted” counts questions on which pair construction was tried; “accepted” counts pairs passing token alignment, the distinct-first-token requirement, and the confidence conditions $M_{\text{clean}} > 0 > M_{\text{corrupt}}$.

direction measures conditional damage. Semantic row, column, and cell corruptions are used instead of Gaussian input noise or an unrelated donor so that each contrast has a known alternative answer, and exact token alignment keeps the patched positions comparable. The primary score is the fraction of the pair-specific first-token logit gap rescued, not a change in generated accuracy. As with all patching, the conclusion is conditional on the corruption and patched site, and a low rescue can reflect redundancy rather than irrelevance [39, 42].

3.4.4 Attribution and Circuit Evaluation

Node-level attribution. Exact patching of every head, MLP, position, and edge is prohibitively expensive on these prompts, so the candidate ranking comes from the gradient-based screens surveyed in Section 2.3. The node-level EAP-IG form is used because it reuses the aligned counterfactual pairs and scores the same margin as the downstream circuit sweep. It is a head/MLP-level adaptation of the node-attribution principle called NAP-IG in MIB, whose reported node evaluations use neuron-level nodes [26]. It scores every head and MLP; only the head subset supplies the ranking for the heads-only faithfulness sweep. It captures clean and corrupt outputs, takes gradients along a corrupt-to-clean embedding path, and scores each node as $(a_{\text{clean}} - a_{\text{corrupt}}) \cdot \partial M / \partial a$ summed over positions, ranking nodes by $|\text{score}|$. The path integral is a five-point left Riemann sum: gradients are taken at interpolation coefficients $\alpha \in \{0, 0.2, 0.4, 0.6, 0.8\}$ and averaged, so the corrupt endpoint is included and the clean endpoint ($\alpha = 1$) is not. Ranking stability at higher integration resolutions was not evaluated; the independently selected named-head ranking, which traces nearly the same faithfulness curve (Section 6.5), provides an indirect robustness comparison. Beyond quadrature, this is a path-dependent approximation to marginal effect rather than a direct intervention or necessity test. Axis-specific signed scores are averaged before taking the absolute value, so opposing effects can cancel; the final ranking also discards sign, and redundant nodes can rank low.

Direct-logit attribution. Direct-logit attribution uses the additive residual-stream decomposition of transformer circuits [7, 39]. At the final position, each head’s output is mapped through its output matrix, adjusted by the final LayerNorm scale, and projected onto the first answer token’s unembedding direction. This identifies heads that directly write toward the answer logit,

but omits effects mediated by later components and is observational rather than causal. It is used only to nominate a writer tier, whose necessity is tested separately by ablation.

Interchange-faithfulness sweeps. Circuit work distinguishes identifying plausible components from showing that the selected set preserves the behavior [9, 39]. Following MIB’s faithfulness-versus-size protocol [26], the clean token sequence is run while every non-circuit head output is overwritten with its cached value from the aligned corrupt run. Circuit heads and all MLPs are left unpatched and recomputed on the resulting hybrid residual; they therefore need not equal their original clean activations. If $M_{K,i}$ is the resulting margin for pair i , the reported faithfulness is

$$f(C_K) = \frac{1}{N} \sum_{i=1}^N \frac{M_{K,i} - M_{\text{corrupt},i}}{M_{\text{clean},i} - M_{\text{corrupt},i}}.$$

I take interchange as primary because each replacement was realized on an example-specific semantic counterfactual. The all-head endpoint is 1 by construction and the zero-head endpoint is 0 empirically; zero and pooled-mean replacements pose different questions, and the pooled-mean variants are empirically degenerate here. The first-token margin provides a continuous, pair-normalized outcome, while generation accuracy remains the separate necessity outcome. The sweep uses exactly 150 pairs from each corruption axis. Thus its primary estimand is an equal-weight four-axis macro-average, not an estimate under the task’s unknown natural distribution of corruptions. Axis-specific values are reported with the pooled curve to expose sensitivity to alternative axis weights. The sweep uses the model-specific grid $K \in \{0, 1, 2, 5, 12, 16, 26, 51, 102, 205, 512, H\}$, where $H = 1024$ for Llama and $H = 784$ for Qwen. Quoted K^* sizes are linear interpolations between grid points, not measured minima; the named-first crossings are additionally bracketed by evaluations at two intermediate sizes, run for the minimality experiment and reported with the curves. Every value is conditional on the declared ablation flavor, metric, ranking, and heads-only circuit class [25].

3.4.5 Candidate-Head Selection

The named-head rosters are selected from the observational routing maps, independently of the causal experiments, by a rule with fixed thresholds: the hub is the head with the best joint rank in column-name-to-header and row-index-to-header routing (required top-5 in both); the column tier is the top four by column-name routing, excluding the hub; the row tier the top four by row-index routing, excluding the hub and the column tier; the retrieval tier the top three by final-token correct-cell attention. The rule’s thresholds were set with reference to the Llama maps and applied unchanged to Qwen’s, so each family’s roster comes from its own observational scores and the cross-family comparison uses no head-index transfer. One Llama row-tier slot departs from the rule: on the Instruct maps the rule selects L5H5, while every Llama experiment uses L14H17 (the head the same rule selects on the base-model maps). The rosters reported throughout are therefore the *tested* rosters, rule-selected at all twelve of Qwen’s slots and at eleven of Llama’s, with the twelfth filled from the base-model maps. L5H5 was never ablated, patched, or placed in a circuit, so nothing measured here is evidence about it, and the Llama row tier’s effects are properties of the four heads actually tested rather than of the rule’s output.

Table 3.3 lists both tested rosters with each head’s selection score; Appendix B gives the full top-ten rankings the tiers were drawn from, with tested-roster membership marked.

role	Llama Instruct		Qwen Instruct	
	head	score	head	score
hub	L8H11	0.779 / 0.649	L15H20	0.890 / 0.579
column matching	L14H19	0.673	L14H0	0.878
	L8H1	0.647	L8H27	0.834
	L4H16	0.552	L8H22	0.829
	L10H2	0.414	L5H19	0.757
row matching	L5H8	0.291	L8H24	0.659
	L5H11	0.258	L4H9	0.372
	L5H9	0.222	L19H20	0.308
	L14H17*	0.193	L4H11	0.256
retrieval	L17H24	0.582	L23H11	0.760
	L16H1	0.533	L22H7	0.688
	L24H27	0.501	L22H1	0.677

Table 3.3: The tested rosters and each head’s raw selection score, from the Instruct observational maps of each family (markdown, zero-shot, rowid; 20,100 questions). Scores are mean attention mass on the tier’s selection route: column tier, column-name token to target header; row tier, row-index token to target index cell; retrieval tier, final token to correct cell; the hub reports its column- and row-routing masses. *The selection rule picks L5H5 (0.303) here; every Llama experiment instead uses L14H17, the head the same rule selects on the base-model maps, which ranks tenth (0.193) on the Instruct map.

The attribution ranking is computed on the full pair banks and the faithfulness sweeps are evaluated on 150-pair-per-axis subsets of the same banks, so the attribution-ranked faithfulness curves are in-sample with respect to head selection and may be optimistic; no held-out pair bank was constructed. The observationally selected named-first ranking does not share this dependence and traces nearly the same curve. A held-out replication is listed in Chapter 9 as future work.

3.5 Reproducibility

All models are evaluated in half precision (fp16) through TransformerLens 2.17.0 on PyTorch 2.8.0 [28, 31]; gradient-based attribution runs in bf16. Runs execute on single NVIDIA A100 80 GB GPUs on a SLURM cluster. Question sampling, mean-activation sampling, and pair construction use a fixed seed (42); the random-head ablation control uses an independent seed, and the random faithfulness rankings use three seeds whose spread is reported as a range. Several causal samples are deterministic corpus prefixes rather than random table samples: mean ablation uses the first 200 sorted files, QK/OV decomposition the first 125, and patching scans sorted files until each axis quota is filled. The behavioral grid uses the full corpus subject to context length. These sampling choices limit population-level generalization beyond the observed tables.

Chapter 4

Behavioral Results

This chapter reports the behavioral results that the mechanistic chapters explain: the formatting levers that move accuracy, the shift from row counting to row matching, and the dominant failure modes. The row-index column is both the largest single lever on accuracy and the manipulation the later chapters use to study how the model finds rows.

4.1 Accuracy Across Formats and Prompts

Figure 4.1 summarizes exact-match accuracy across the full condition grid. Two coarse regularities hold everywhere. First, **markdown is easier than CSV**: markdown beats CSV in every one of the twelve matched Llama cells, by 0.06 to 0.23 (for example, Instruct zero-shot with a row index scores 0.912 on markdown against 0.814 on CSV). The advantage is not a pooled artifact; it holds cell by cell, and all four matched no-index comparisons carry table-level cluster-bootstrap intervals well clear of zero, computed on the 10,020 questions each markdown and CSV pair shares across 1,295 tables: base zero-shot +0.164 (95% CI [+0.151, +0.176]), base two-shot +0.056 ([+0.045, +0.067]), Instruct zero-shot +0.108 ([+0.098, +0.119]), and Instruct two-shot +0.150 ([+0.138, +0.162]; Section 3.4.1). Earlier 6,700-question runs of the base two-shot and Instruct zero-shot conditions give +0.062 and +0.106 on their own smaller shared subsets, replicating these two cells on independent question samples. Second, **scaffolding helps but does not equalize the families**: without any scaffolding, Qwen starts markedly higher than Llama (Qwen base zero-shot on plain markdown scores 0.580 against Llama’s 0.439), a base-model gap that Chapter 7 connects to training data. Instruction tuning and few-shot examples usually narrow this gap, although Llama narrowly overtakes Qwen in two base-CSV two-shot indexed cells (rowid 0.763 versus 0.749; randidx 0.727 versus 0.723). The cross-family ordering is therefore a broad tendency rather than an invariant.

4.2 Effects of Explicit and Reversed Row Indices

The largest formatting effect comes from an explicit row-index column (Figure 4.2). Adding one lifts accuracy by 0.16–0.33 on markdown and 0.18–0.28 on CSV, in every cell of the grid and in both families: Llama base zero-shot rises from 0.439 to 0.768 (a +0.329 lift, 95% CI

Cell-lookup accuracy across families, formats, prompts, and row-index conditions

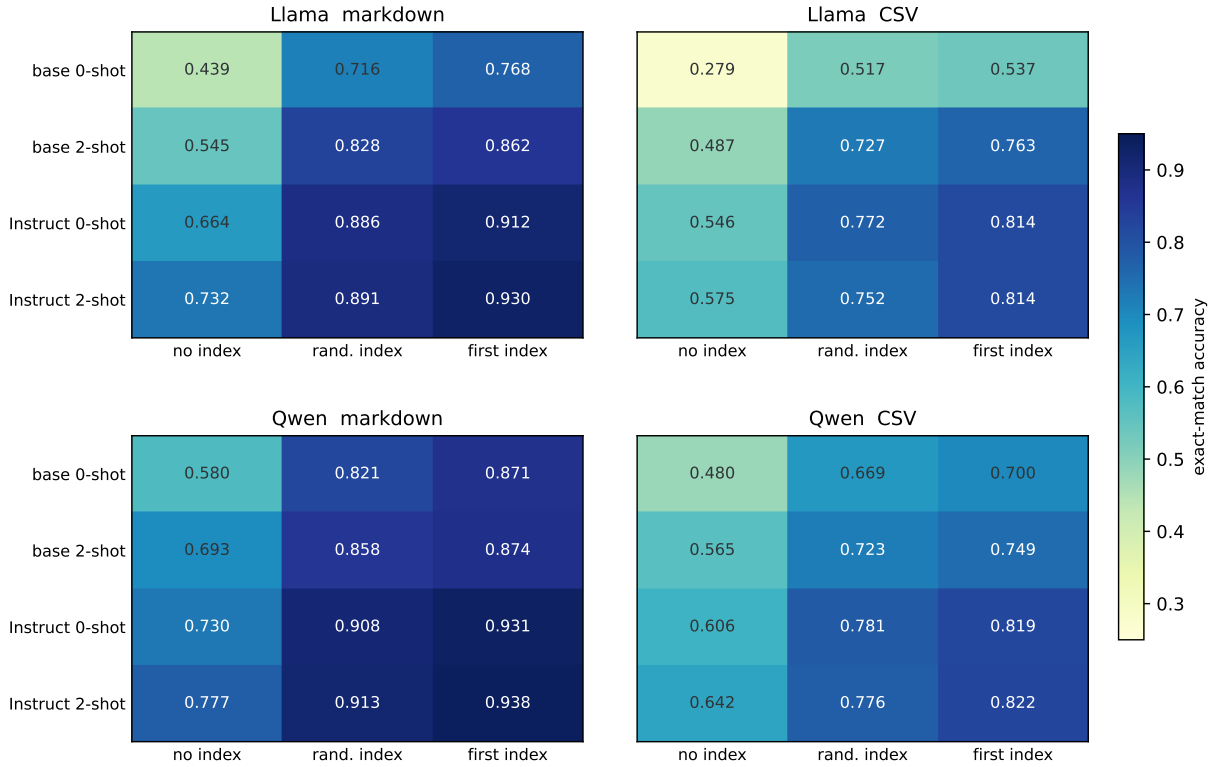


Figure 4.1: Exact-match accuracy across family, tuning, format, shots, and row-index condition ($n \approx 20,100$ questions per cell). Markdown exceeds CSV in every matched Llama cell, and within each format the row-index column (rowid) is the single largest lever.

[+0.317, +0.341]), and Qwen base zero-shot from 0.580 to 0.871 (+0.29, matching Llama’s +0.33). The error analysis identifies what the index fixes. The dominant failure without an index is `off_by_row` (right column, wrong row): it accounts for 62.1% of Llama base markdown errors and 70.7% of Instruct markdown errors with no index, and collapses to 23.5% and 12.0% once the index is present (Figure 4.3). (The bucket counts *any* wrong-row answer, not only adjacent rows.) The share falls monotonically from none to randidx to rowid in thirteen of the fourteen model $\times$ format $\times$ shots combinations whose error-bucket records cover all three row-index conditions (the two Llama base markdown combinations lack per-example records for their indexed cells); the exception, Llama base CSV zero-shot, inverts by 1.8 pp and is the weakest cell in the grid (0.279 accuracy).

Falling shares alone would not establish that the index *converts* those particular errors: the same aggregate movement could be produced by fixing different questions while introducing new row errors elsewhere. Because the two conditions sample the same questions, the transition can be measured directly. Pairing all 20,100 questions of the Llama base zero-shot markdown cell between its no-index and rowid runs and re-classifying each under both, 76.9% of the questions that were `off_by_row` without the index are answered correctly with it (5,386 of 7,004;

88.9%, or 4,249 of 4,777, in the Instruct zero-shot pair). The exchange is almost entirely one-directional (6,832 questions are gained against 211 lost), and 78.8% of everything gained comes out of the `off_by_row` bucket alone. Of the 1,618 original wrong-row errors that are *not* repaired, 877 (54.2%) remain wrong-row rather than migrating to another failure; equivalently, these are 12.5% of the original 7,004 wrong-row errors. The row-index column therefore repairs the specific failure it was predicted to repair, and almost all of its benefit comes from that one bucket. The residual errors shift toward `not_in_table` and `wrong_col_same_row`, which the mechanistic chapters trace to the retrieval stage rather than to row finding.

One robustness check guards the reading of exact match as “found the cell.” Because questions are sampled from all valid cells, some targets are empty (3.5%) or non-unique—their answer string also occurs in another cell of the table (38.8%)—and for those an exact match need not prove the right coordinate was located. Restricting to the 57.7% of questions whose target is unique and non-empty leaves the row-index result intact and in fact sharpens it: on this subset Llama base zero-shot markdown scores 0.399 without the index and 0.776 with it, a +0.377 lift (95% CI [+0.362, +0.393]) against +0.329 on the full set. Removing ambiguous targets lowers the no-index accuracy more than the indexed one (some no-index “successes” were coincidental string matches), so the row-index effect holds under the stricter metric.

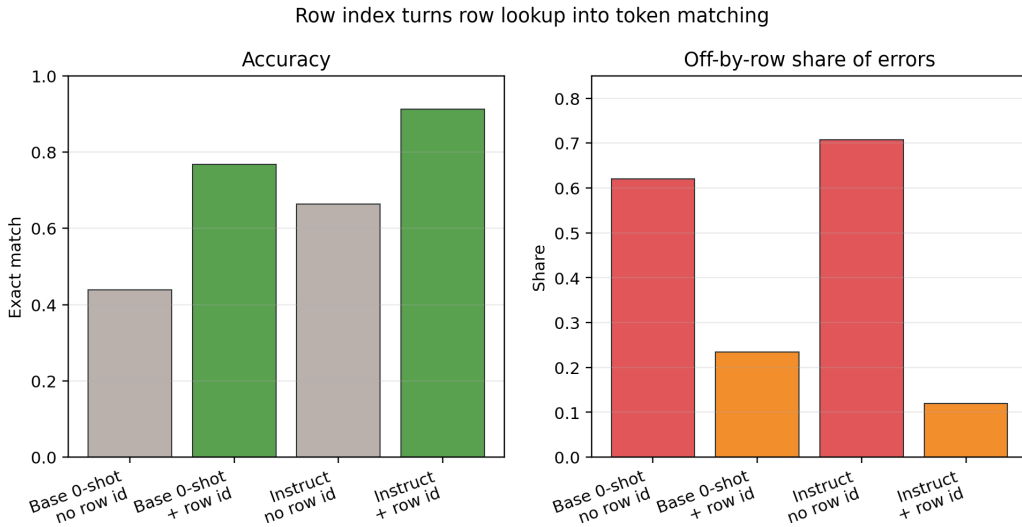


Figure 4.2: Accuracy with and without a row-index column across the markdown grid, in both families.

The reversed-order condition separates two accounts of how the index helps. If the model found rows by counting positions, an index that merely relabels positions could be ignored; if it matches the requested number against the index column’s contents, the benefit should survive reversing the rendered order, in which the index reads $N, \dots, 1$ top-to-bottom while “row N ” still refers to the row labeled N . Zero-shot, the row-index lift largely survives reversal (Figure A.1, Appendix A): Llama base retains 66% of the none→rowid lift (0.655 against 0.439/0.768), Llama Instruct 79% (0.860 against 0.664/0.912) and Qwen Instruct 90% (0.912 against 0.730/0.931). In Llama the ordering $\text{rowid} > \text{randidx} > \text{revidx}$ holds at both shot levels, reversal costing more

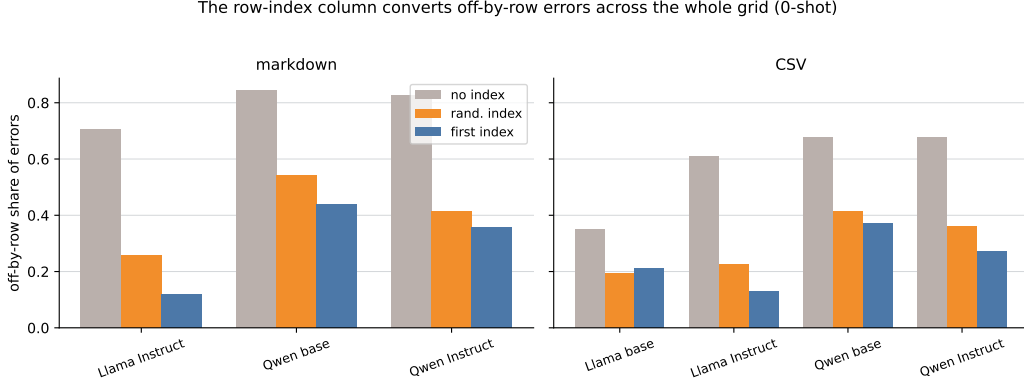


Figure 4.3: Failure-bucket composition with and without a row-index column, Llama markdown. The `off_by_row` share is what the index removes; the residue is `not_in_table` and `wrong_col_same_row`.

than relocating the index column; in Qwen Instruct the two are level (0.912 reversed against 0.908 relocated), so reversal costs no more there than moving the index. These results are consistent with content-based row lookup even when the label order contradicts physical order; the row-index swap of Chapter 6 (axis d) supplies causal evidence for that account, on its selected aligned-pair subset rather than the full behavioral distribution.

Two-shot, in contrast, falls *below* zero-shot under reversal, and does so in both families (Table 4.1). The comparison uses a ten-row example bank throughout: two example tables with targets at rows 2 and 8, which render at positions 9 and 3 once reversed, so both demonstrations display the reversal rather than leaving a mid-table target in the same place either way. Because the demonstrations are rendered under the same reversal as the table, the drop cannot be a mismatch between monotonic examples and a non-monotonic table, and because their targets move under reversal it is not an artifact of examples too short to show the mapping. Writing Δ for the two-shot minus zero-shot change, the direct test that the few-shot penalty is specific to reversal is the interaction $\Delta^{\text{revidx}} - \Delta^{\text{rowid}}$, estimated with a table-clustered paired bootstrap in which one table draw is applied to all four cells: -0.095 (95% CI $[-0.104, -0.086]$) for Llama base, -0.064 ($[-0.069, -0.058]$) for Llama Instruct and -0.048 ($[-0.054, -0.043]$) for Qwen Instruct. Adding examples costs 0.038 – 0.061 under reversal while moving the ordinary rowid condition by -0.010 to $+0.057$, and the interval on the difference excludes zero in every model with a measured reversed zero-shot baseline. One qualification is that every two-shot cell uses the same fixed two-example bank, so what is established is an interaction between reversed order and this demonstration set; replication over sampled banks remains to be done. The cause is not diagnosed here, since the circuit measurements are made zero-shot; Section 9.4 sets out the experiments that would settle it.

model	rowid		revidx	
	0-shot	2-shot	0-shot	2-shot
Llama base	0.768	0.825	0.655	0.617
Llama Instruct	0.912	0.915	0.860	0.799
Qwen base	0.871	0.861	—	0.775
Qwen Instruct	0.931	0.932	0.912	0.865

Table 4.1: Exact-match accuracy with an ordinary row index (rowid) and a reversed one (revidx), markdown, $n \approx 20,100$ per cell. Two-shot cells use the ten-row example bank. Adding examples moves the ordinary condition by -0.010 to $+0.057$ but costs 0.038 – 0.061 under reversal; table-clustered bootstrap intervals for the deltas and their interaction are given in the text. The Qwen base reversed zero-shot cell was not complete in time. Appendix A reports the same comparison on the original three-row bank.

4.3 Stability to Index-Column Position

If rows are anchored by index content rather than position, moving the index column should cost little. It does: randomizing the index’s column position (randidx) costs only 0.02 – 0.06 relative to placing it first (a rowid-minus-randidx difference of $+0.052$ for Llama base zero-shot markdown, 95% CI $[+0.046, +0.058]$), across the entire grid and both families. The invariance is visible not just in accuracy but in the internal routing: the ranked per-head attention maps are nearly identical between the fixed- and randomized-position conditions, with a median rank correlation of 0.966 for row-number-to-header routing and 0.9996 for column-name routing (Figure 4.4). The row-matching route therefore finds the explicit index landmark wherever that *index column* sits, which licenses studying the first-column (rowid) regime causally without treating index-column location as a confound. This intervention moves the row landmark, not the queried data field; it does not imply that arbitrary target columns are equally easy in tables of every width.

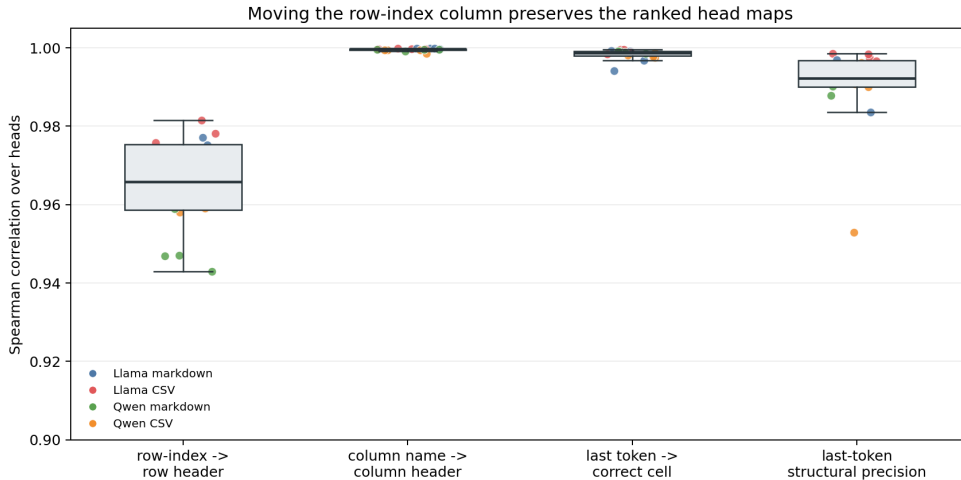


Figure 4.4: Accuracy and ranked per-head routing maps under randomized row-index column position (randidx), against first-column placement.

4.4 Failure Modes: Off-by-Row and Column Geometry

The errors fall into two families. The first is row tracking: the `off_by_row` errors of Section 4.2, which the row-index column addresses. The second concerns the column axis. A marginal plot makes high column indices look difficult, but those indices exist only in wide tables, so width is confounded with position; the analysis below holds width exactly fixed to separate the two.

Width affects even the earliest columns. When the query is restricted to the first three data columns, median accuracy falls from 0.769 in the 8–10-column bin to 0.592 in the ≥ 11 bin; the within-condition change has median -0.149 and is negative in all 33 conditions. Width-related table difficulty therefore reaches the left edge as well as the right. At a fixed exact width the first-three and last-three column bands share columns below width six, so the position contrast is read from width six upward (Section 3.4). There, the last three columns are only 0.009–0.031 below columns 0–2 at widths 6–8 (98–357 source tables per width), then trail by 0.085 and 0.105 at widths 9 and 10 (38 and 23 tables). Width 11 gives -0.063 over 14 tables. Widths 12–14 give -0.119 , -0.145 , and -0.080 , but rest on 3, 7, and 16 tables respectively (Figure 4.5); repeated question records inflate their nominal sample sizes without adding independent tables, so those three points are exploratory and possibly template-specific. The evidence for the position gradient rests on widths 9–11, each with tens of independent tables. In the pooled ≥ 11 bin the contrast between columns 0–2 and columns 9+ has median -0.162 across conditions and is negative in 32/33, although that binned contrast still mixes exact widths. In sum, queried-column position costs little through roughly eight data columns and becomes material toward the right edge of wider tables; it is not a monotone penalty at every width.

The same exact-width analysis also revises the apparent cutoff. Among lexical-class tables (those with fewer than half of their header cells numeric-like or blank), pooled accuracy is 0.705–0.765 from widths 4 through 11, with no downward gradient. Widths 12 and 13 fall to 0.661 and 0.609 but contain only two and five tables. Width 14 falls to 0.285 over 15 tables, 14 of which share the same monthly “Month–Jan–...–Dec–Year” schema. Consequently the low pooled ≥ 11 bin is a real description of this evaluation corpus, but it cannot identify either a universal ten-column threshold or a smooth causal effect of adding columns.

Header form supplies a second compositional signal. Tables whose headers are at least half numeric-like or blank score 0.486 against 0.718 at widths 5–7 and 0.454 against 0.719 at widths 8–10. The gap is only 0.026 in the ≥ 11 bin, where lexical-class tables also decline. The numeric/empty groups contain only 6, 8, and 7 underlying tables in the three bins. Their moderate-width deficit is consistent with the lexical sensitivity localized in Section 7.2, but header class was not intervened on and may instead mark hard source or template families.

An additive logistic regression gives a coarse summary of the same data, not an independent effect estimate. Per-question exact match was regressed jointly on z -scored absolute column index, data width, and target row index using `scikit-learn`’s default L_2 penalty ($C = 1$) [32], separately for Llama base and Instruct $\times$ markdown and CSV without a row index (6,700 questions per fit). Width coefficients are -0.11 to -0.13 , absolute-column coefficients -0.04 to -0.06 , and row-index coefficients -0.43 to -0.69 . The fit has no table-clustered uncertainty, omits the width-by-position interaction, and can absorb header and table-family composition into its width term. It therefore describes average association under one specification rather than establishing which geometric variable causes the errors.

The marginal “far-right” pattern thus mixes table width, within-table target position, and corpus composition. Section 5.6 asks how attention changes within the same coarse widest-table bin, and Chapter 8 turns the qualified result into provisional rather than threshold-based guidance.

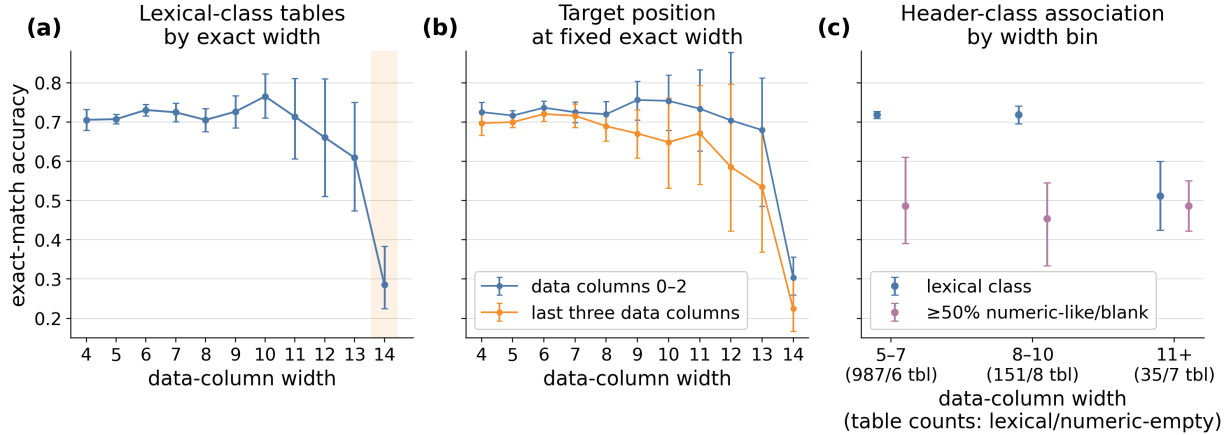


Figure 4.5: Column geometry pooled across 33 **none/rowid** conditions. Left: accuracy for lexical-class tables (fewer than half of their header cells are numeric-like or blank) is broadly similar at widths 4–11; widths 12 and 13 are sparse, containing only two and five tables, while 14/15 lexical-class width-14 tables share a monthly schema. Middle (all header classes): the last three data columns trail columns 0–2 only slightly through width 8, with a larger but non-monotone penalty from width 9. Right: tables with at least half numeric-like or blank headers trail lexical-class tables by 0.232–0.265 at widths 5–10, but not in the ≥ 11 bin. The analyses condition on observed geometry and header form rather than intervening on them; error bars are 95% table-cluster bootstrap intervals.

4.5 Base versus Instruct

The instruction-tuned models look substantially stronger, but most of that advantage is prompt format. Running the Instruct weights through the *base-style* two-shot text prompt—byte-identical to what the base model sees—isolates the weight effect: base 0.862 $\rightarrow$ Instruct 0.876, a lift of only +0.014. Three local contrasts, all in the Llama markdown row-index condition, measure the changes associated with each factor: prompt **format** (few-shot text $\rightarrow$ chat template) gives +0.054; **shots** (0 $\rightarrow$ 2, holding chat format) +0.018; and **weights** (base $\rightarrow$ Instruct, matched text prompt) +0.014 (Figure 4.6). The three contrasts use different baselines, so they are not an additive decomposition of the Instruct advantage. The pattern motivates the mechanistic split of Chapters 5–6: the matching machinery is largely pretrained and shared across tuning, while instruction tuning re-weights the late retrieval and read-out stage.

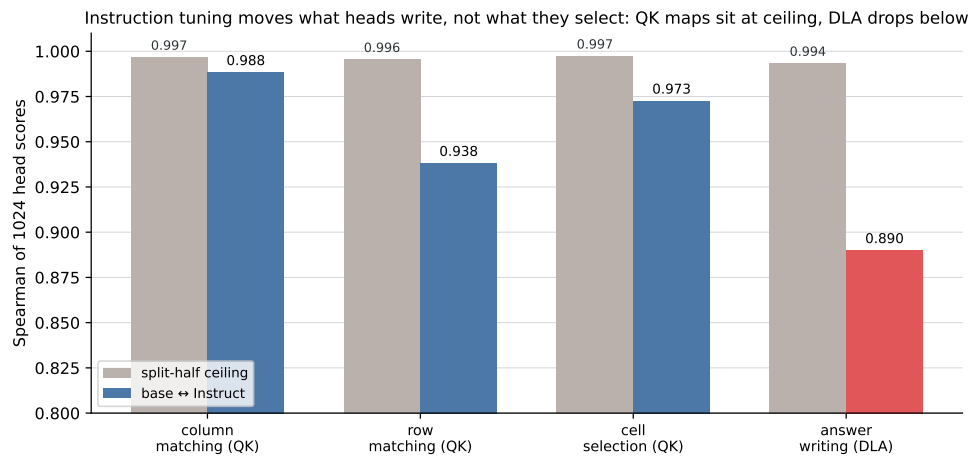


Figure 4.6: Three local contrasts associated with prompt packaging, shot count, and tuned weights, measured at the row-index condition. The three use different baselines and do not sum to the Instruct advantage; the largest of them accompanies prompt packaging.

Chapter 5

Observational Evidence for Matching and Retrieval

This chapter builds the observational case for a two-stage pipeline from attention alignment, question-token routing, and layer-depth profiles, and relates attention strength to behavior across conditions. Everything here is correlational: attention mass, not intervention. Its purpose is to locate the candidate heads and establish the depth ordering that Chapter 6 then tests causally. The evidence supports a *matching* stage in the early and middle layers that binds the question’s column name and row index to landmarks in the table, followed by a *retrieval* stage in the late layers that routes to the target cell and assembles the answer.

5.1 A Two-Stage Depth Profile

Scoring every head for how strongly its attention tracks the table’s structure yields two metrics defined in every condition: a column-matching metric (column-name-to-header routing) and a retrieval metric (final-token-to-correct-cell attention). Row-index routing is analyzed separately on indexed conditions; it is not folded into this cross-condition profile. Averaging each layer’s strongest heads and taking a score-weighted centroid, normalized to $[0, 1]$ so that Llama’s 32 layers and Qwen’s 28 are comparable, the matching centroid sits at median normalized depth 0.466 and the retrieval centroid at 0.721; the median within-condition gap is 0.234 of model depth (Figure 5.1).

The ordering is consistent: **retrieval is later than matching in all 48 completed conditions** (Llama and Qwen, markdown and CSV, base and Instruct, three row-index levels, zero- and two-shot), the peak retrieval layer is likewise later in every cell, and the ordering survives whether the centroid is computed over the top 5%, 12.5%, 25%, or all heads per layer. This is descriptive replication, not 48 independent samples (Section 3.4.1), but the ordering itself is stable across every condition measured.

The same ordering appears in indexed and non-indexed conditions, although accuracy differs substantially: the no-index conditions sit 0.16–0.33 below the indexed ones (Section 4.2). The common measurement supports the narrower statement that retrieval occurs later than the available column-matching signal in both regimes. It does not establish that a row-matching stage

remains present but fails without an index, because no row-index landmark exists there to score.

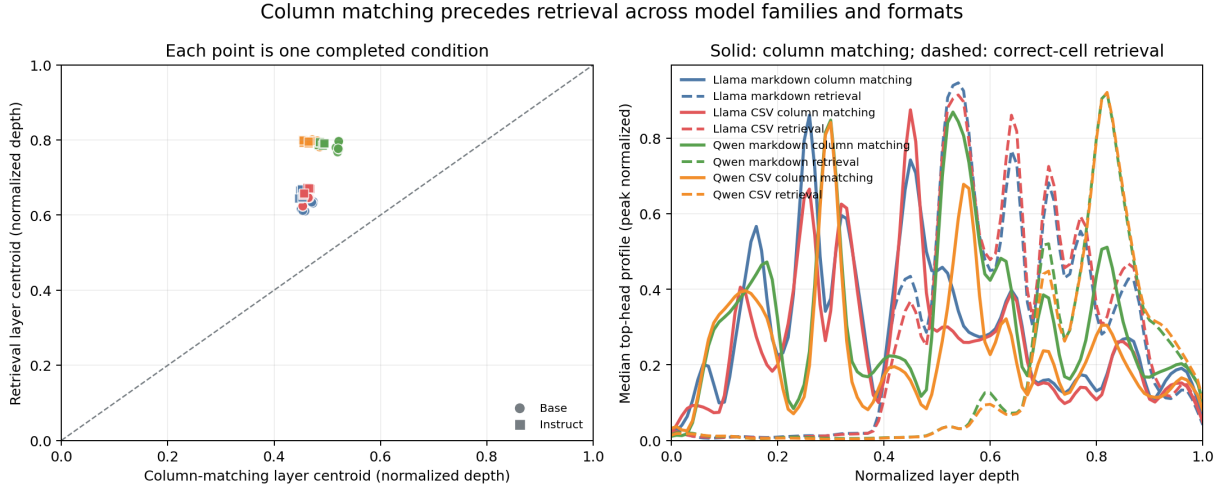


Figure 5.1: Normalized column-name-to-header and correct-cell-retrieval layer centroids across all 48 completed conditions. The same column-matching metric is used in indexed and non-indexed conditions. Every point lies above the diagonal.

5.2 Within-Table Column Alignment

The matching stage first has to establish where the columns are. Scoring each head’s attention against a causal-uniform baseline isolates a tight set of column-aligned heads in the early and middle layers: in Llama Instruct, L8H11, L4H16, L14H19, and L10H2 carry the strongest column-name-to-header and pre-question column-alignment signal (peak alignment score ≈ 0.61 ; Figure 5.2, left). These same heads recur across the alignment, column-name routing, and row-index routing analyses rather than splitting into separate clusters; the stages separate by depth, not by disjoint head sets. Consistent with the behavioral base-versus-Instruct result of Section 4.5, this column-matching machinery is essentially tuning-invariant: the per-head column-name routing map has a base-to-Instruct rank correlation of 0.951 in Llama and 0.988 in Qwen, and the column- and row-alignment maps are higher still ($\rho \geq 0.99$ in both families), whereas the retrieval metrics move more under tuning (correct-cell mass 0.873 and 0.895, structural precision 0.802 and 0.867; Appendix B). The matching stage is largely pretrained.

Scored separately by axis, the two coordinates are not tracked alike. Column alignment is sparse and strong: a few isolated heads stand well clear of an otherwise flat map, led by L8H11 (0.611) and L4H16 (0.537). Row alignment is weaker by more than half (the strongest head reaches 0.269) and far more diffuse: its few sharp peaks (L4H25, L4H30) sit in the first handful of layers, the rest is a broad low band over the early and middle layers, and the per-layer mean is positive through layer 16 and essentially vanishes above it. The two maps also barely overlap: the leading column heads carry essentially no row alignment ($|\cdot| < 0.02$), and the leading row heads none of the column signal. This is a statement about attention *geometry*, which cells of the

table a head reads, and not about which question tokens it binds: L8H11 has no row alignment at all, yet performs the model’s strongest row-index-to-header routing (Section 5.3). Tracking the table’s row structure and binding the question’s row key to a landmark inside it are separate operations, and it is the second that the matching stage runs on.

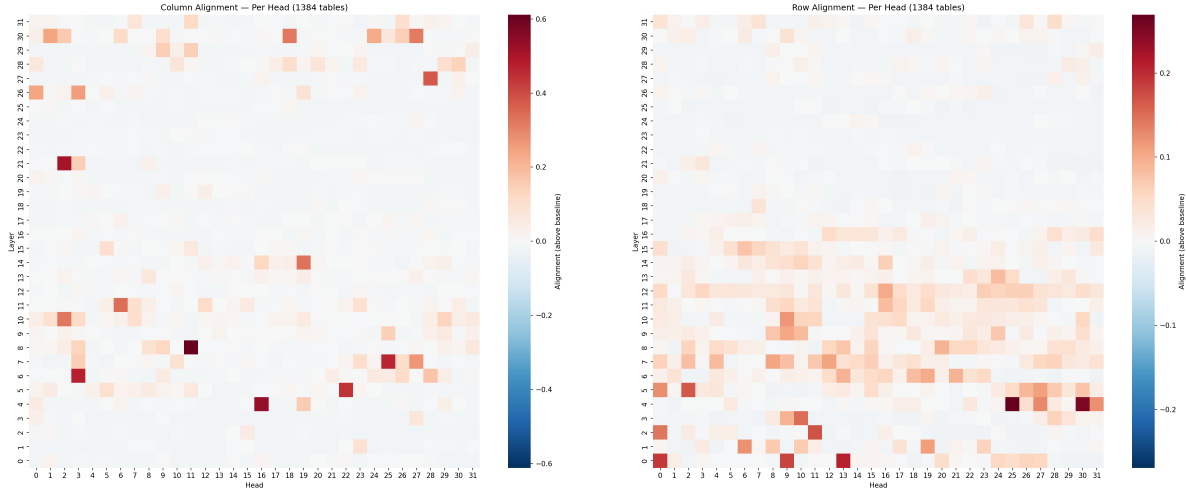


Figure 5.2: Per-head attention alignment against a causal-uniform baseline (Llama Instruct, markdown, row-index, 1,384 tables; the alignment maps need no question and so run over the full corpus, before the duplicate-header exclusion that leaves the 1,340 tables used everywhere questions are generated, Section 3.2). Left: column alignment concentrates in a small set of early/middle heads against a near-flat background. Right: row alignment is less than half as strong and spread broadly over the early and middle layers rather than concentrated. Note the differing color scales.

5.3 Question-to-Table Routing

Within the matching stage, one head carries the strongest routing on both axes. In Llama, L8H11 is a *hub* that performs both column-name-to-header and row-index-to-header routing (column-name score 0.78, row-index score 0.65). That routing is content selective. Figure 5.3 contrasts the attention of different question tokens: the *column-name* token and the *row-index* value bind strongly to their corresponding headers and index cells, while the matched generic keywords do not: “row” reaches only 0.06 at the row header against the index value’s 0.65, and “column” only 0.20 at the target header against the column name’s 0.78, both content peaks falling at L8H11 itself. The model matches the requested *index value* against the contents of the index column, not the word “row” or the row’s position. This parallels the behavioral content-anchoring result (Sections 4.2–4.3): among the row landmarks tested, the explicit index value is the one this routing binds to.

The functional division of labor replicates across families, though not the head coordinates. Applying the same behavior-based selection rule to Qwen’s attention maps recovers a hub (L15H20)

that does both kinds of matching and its own early/middle matching cluster: the same split into column matching, row-index matching, and a joint hub, at different (layer, head) positions. The same depth-ordered split recurs in both tested families; the wiring (which layer, which head) does not. Appendix B collects the full per-head maps behind this chapter’s summaries, Qwen’s roster-defining maps, and the top-ten rankings for every selection metric with tested-roster membership marked.

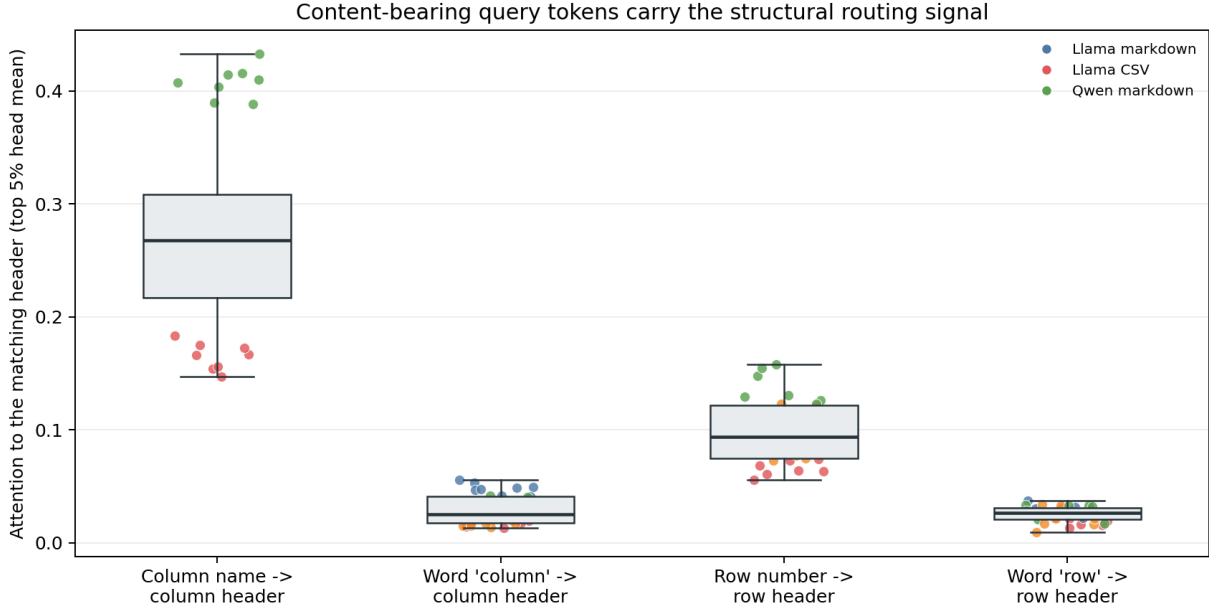


Figure 5.3: Attention from specific question tokens to table landmarks. The column-name token and the row-index value bind to their header and index cell; generic “row”/“column” keywords do not.

5.4 Attention to the Target Cell

The strongest retrieval scores occur in the late layers and are spread over more heads than the matching scores. In Llama, the heads carrying the strongest final-token-to-correct-cell attention and structural precision sit around layers 16–24: L17H24, L16H1, and L24H27 lead, with a band of weaker contributors and a set of gain heads near layer 22 that strengthen under instruction tuning. In Qwen the analogous heads peak near layers 22–23 (Figure 5.4). Unlike the column-matching heads, no single retrieval head dominates: correct-cell mass tops out around 0.58 and is spread over a dozen heads. The co-localization view (Figure B.5) shows the retrieval metrics overlapping on a shared late-layer pool rather than isolating cleanly. This diffuseness is a fact about attention mass, and must not be read as substitutability: causal importance concentrates in exactly this stage, whereas the *matching* heads, observationally the most concentrated, turn out to be the individually substitutable ones. Observational spread and causal concentration are different quantities, a dissociation Chapter 6 makes precise.

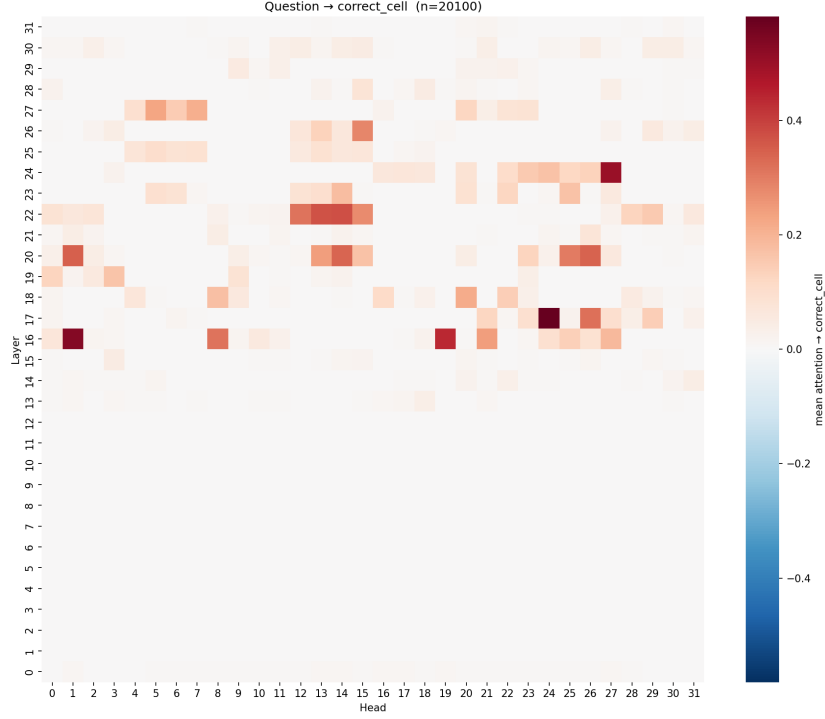


Figure 5.4: Per-head final-token attention to the correct cell (Llama Instruct, markdown, zero-shot, rowid, $n = 20,100$). The strongest heads sit in the late layers and no single head dominates. Appendix B gives the structural composition of this attention, the corresponding Qwen map, and the metric co-localization maps for all four models.

5.5 Relating Attention to Behavior

The last analysis relates these attention signals to accuracy across the 48 conditions. The mean correct-cell mass of the top 5% of heads correlates with exact-match accuracy at Spearman 0.94 across the twelve Llama-markdown conditions, 0.91 across Llama CSV, 0.82 across Qwen markdown, and 0.86 across Qwen CSV (Figure 5.5). The top 5% is reselected within each condition, so these are correlations between each condition’s strongest heads and its accuracy, not the trajectory of one fixed head set. Structural precision (the share of a head’s structurally directed mass that lands on the correct cell) tracks accuracy about as closely as raw correct-cell mass does (0.94 Llama markdown, 0.99 Llama CSV, 0.83 Qwen markdown, 0.85 Qwen CSV), so both retrieval-side readings of the same attention agree. Column-name routing, by contrast, does not track accuracy in three of the four panels (-0.15 to -0.32) despite its head map being the most stable of all. The pattern is consistent with lower variation in column matching: it is strong in every markdown condition, so its variation no longer separates easy prompts from hard ones. The exception is Qwen CSV ($+0.72$), the panel where column matching is weakest and varies most. Across panels, accuracy differences track the retrieval-side metrics more closely than column matching.

Row-index-to-header routing has to be read on a narrower base. The metric scores atten-

tion from the question’s row number to the cell at (target row, index column), and in the **none** conditions no index column exists, so the implementation falls back to the first *data* column of the target row. That fallback is a positional proxy for a landmark the prompt does not contain, not a measurement of row-key routing, so the primary result is read on the eight indexed (**rowid**, **randidx**) conditions per panel, where the landmark is real: the correlations are 0.38 Llama markdown, 0.64 Llama CSV, 0.38 Qwen markdown, and 0.29 Qwen CSV—positive in all four panels. On eight points apiece none of the four is individually distinguishable from zero (exact permutation $p = 0.08$ to 0.46). Row routing therefore covaries with accuracy in the expected direction without being established as a cross-condition predictor. Pooling the proxy conditions back in would raise the same correlations to 0.76–0.80, which is why those are not the values reported here. The retrieval-side metrics survive the restriction: on the same eight-condition base they run 0.76–0.98, each significant at $p < 0.05$.

These are aggregate, cross-condition correlations over related conditions within a panel, not evidence that attention causes a correct answer on any individual example. And because the top 5% is reselected inside each condition, what they establish is that each condition’s *strongest* retrieval attention covaries with its accuracy—not that any fixed set of named heads does. Whether the named heads themselves carry the effect is the question the interventions of Chapter 6 test.

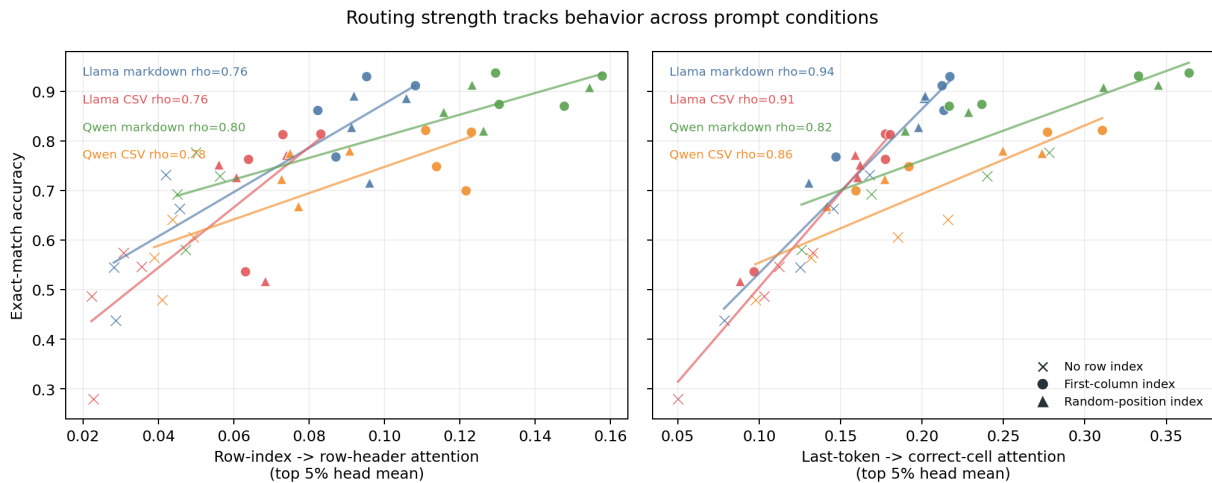


Figure 5.5: Exact-match accuracy against correct-cell retrieval attention (right) and row-index-to-header routing (left) across the 48 conditions. Fits are descriptive within each family/format panel, not per-example causation. In the **none** conditions the row-header target is the first data column standing in for an absent index column, a positional proxy rather than a row key; the primary routing correlations in the text are restricted to the indexed conditions.

5.6 Attention in Wide Tables

The widest-table association of Section 4.4 raises the question of which stage gives way; conditioning per-head attention on table width answers it only partly. Re-aggregating the per-example question-to-cell attention by the coarse width bins of Section 4.4 (width counted in *data* columns,

so a table falls in the same bin whether or not a row index was added to it) covers the 44 grid conditions whose per-example attention was retained. Accuracy falls in the ≥ 11 bin in every one of them, and so does every attention quantity that tracks the target: correct-cell mass in the top 5% of retrieval heads, structural precision, and column-name-to-header routing, each in 44/44 conditions. These are repeated configurations of essentially the same sparse table subset, not 44 independent samples of wide tables. Within that coarse partition the target-related signals decline together (Table 5.1): the median step from 5–7 to 8–10 columns is small, whereas the step into the ≥ 11 bin is much larger for the matching signal as well as the retrieval one. This co-decline does not localize the association to one stage. Part of the column-header decline may be mechanical, since the target header is one of more headers, but differences in shared source difficulty and header form could also accompany low values in every measured signal. The one quantity that does not decline in the top bin is row-header routing, which falls across the middle step and then changes by +2.4% at the median (down in only 20/44 conditions).

Nor does this binning locate a behavioral breakpoint. The exact-width reanalysis in Section 4.4 finds lexical-header accuracy broadly flat through width 11, only two and five lexical tables at widths 12 and 13, and a width-14 cliff concentrated in one monthly-table schema. The attention result therefore says that routing, retrieval, and accuracy are all weak on this corpus’s pooled widest-table subset; it does not show that adding an eleventh column causes them to break.

What the matched formatting contrast does show is that the row-index column blunts the widest-bin association, and that this depends on the serialization. On markdown the accuracy loss in the widest bin falls from a median 28.8% without an index to 11.7% with one; on CSV the same comparison moves only from 45.7% to 33.6%. The index also has the larger mitigation when placed first: randomizing its position gives 15.3% on markdown and 45.1% on CSV, so the position invariance of Section 4.3 weakens in the widest tables even though it holds across the grid as a whole. Two limits bound all of this. Per condition, the widest bin is thin (630 questions over 42 tables) and heterogeneous, spanning 11 to 25 data columns; its exact-width tail is concentrated in a few templates. Moreover, the measurement conditions attention on observed width and table identity rather than intervening on either, so it cannot separate geometry from source-family composition.

quantity	median change per width step	
	5–7 $\rightarrow$ 8–10	8–10 $\rightarrow$ ≥ 11
exact-match accuracy	–2.8%	–34.3%
correct-cell attention	–6.4%	–24.2%
structural precision	–5.3%	–19.7%
column-header routing	–2.3%	–27.4%
row-header routing	–17.5%	+2.4%

Table 5.1: Median change between coarse width bins across the 44 grid conditions with per-example attention retained, with attention read over the top 5% of heads per metric (heads fixed on the pooled examples). Width counts data columns, so indexed and non-indexed conditions bin identical tables. Accuracy, retrieval attention, and column matching all decline in the same sparse ≥ 11 subset, so the association does not localize to one stage; exact-width analysis does not support interpreting the bin boundary as a universal breakpoint.

Chapter 6

Causal Tests of the Candidate Circuit

This chapter uses interventions to test the observational pipeline: which heads the model needs, where counterfactual information can be restored, how large an interchange-faithful circuit is, and what inverse-complement and minimality evidence supports it.

The tests estimate different quantities: mean ablation probes necessity, activation patching measures marginal repair, gradient attribution ranks candidates, and counterfactual interchange evaluates component sets. Their shared definitions are in Section 3.4. The picture they produce is a *matching* stage in the early and middle layers (hub L8H11; column matching L4H16/L14H19/L8H1/L10H2; row-index matching L5H8/9/11, L14H17) that binds the question to table landmarks, a late *retrieval* stage (L17H24/L24H27/L16H1) that routes to the target cell, and a diffuse late write-out tier that is redundant under mean ablation. These twelve named Llama heads form a dense small- K core, but the named-first circuit reaches the declared $f = 0.8$ threshold only in the low hundreds of heads, bracketed between 219 and 274 in Llama and at 180 or fewer in Qwen (Section 6.5).

6.1 Mean-Ablation Necessity

Mean ablation estimates full-generation necessity under a task-average replacement (Section 3.4); the conclusion is necessity under that replacement distribution, not context-independent indispensability. Its effect is read from changes in accuracy and error composition. In the Instruct, markdown, row-index regime (baseline 0.9100), the four column-matching heads have the largest single-stage effect: ablating them drops accuracy by 0.136, against only 0.003 for twelve random heads (Figure 6.1). Each set fails in a characteristic way. Ablating the row-index matching heads (L5H8/9/11, L14H17—the tested row tier, whose fourth slot departs from the selection rule, Section 3.4.5) raises the `off_by_row` share of errors by 19.9 percentage points, from 9.7% to 29.6% among errors. Over all 1600 questions, the corresponding rate rises from $14/1600 = 0.88\%$ to $56/1600 = 3.50\%$. Ablating the column-matching heads costs the most accuracy but, contrary to the naive prediction, the composition among errors does not concentrate on the neighboring column: the `wrong_col_same_row` share falls by 10.4 points while `not_in_table` rises by 13.8 points. The absolute wrong-column rate, however, increases from $56/1600 = 3.50\%$ to $103/1600 = 6.44\%$. Column ablation therefore increases wrong-column

failures but increases total and off-table failures still more. Ablating the retrieval trio costs 0.038 and shifts error composition toward off-table answers as well.

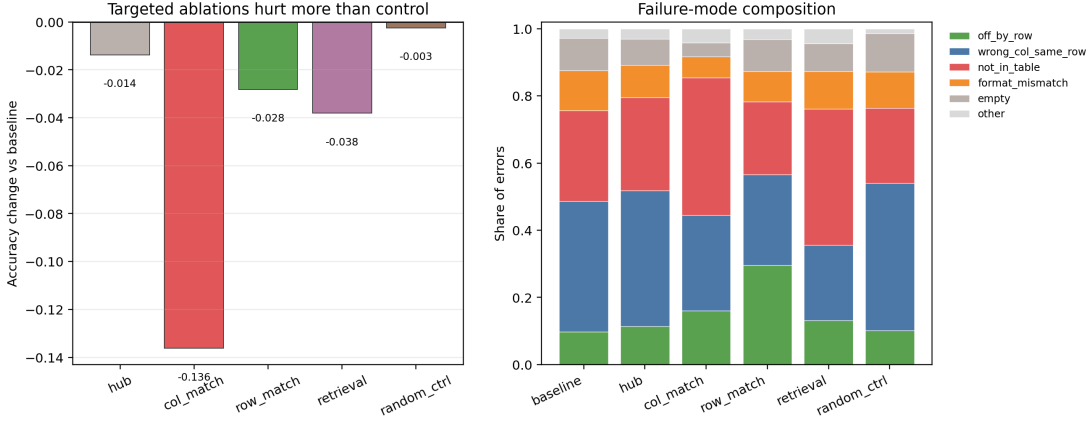


Figure 6.1: Accuracy drop from mean-ablating each stage (Llama Instruct, markdown, row-index), against a twelve-head random control. The specificity control does not hold in the base two-shot regime (Section 6.1).

Two facts qualify these numbers. First, the specificity control does not hold in every regime. Repeating the ablations on Llama base two-shot, where baseline row tracking is already marginal (*off_by_row* 34% of errors), makes the column bottleneck even larger (-0.231) but breaks the row-matching result’s specificity control: twelve random heads now cost 0.041 and raise *off_by_row* as much as the row-matching heads do. Where baseline row tracking is marginal, any residual-stream perturbation tips cases into off-by-row, so the row-tier attribution holds only where it is measured, in the Instruct regime. For that reason I run the circuit-adequacy program from here on in the single regime of Table 3.1. Second, the division of labor replicates across families while the *concentration* does not. Applying the same behavior-based selection rule to Qwen and ablating its functionally-found row-matching heads raises *off_by_row* by 11.0 pp against a -0.9 pp control (the same selective row-anchoring failure at different coordinates), but none of the tested small, functionally selected Qwen head sets is a bottleneck (every Qwen single-stage ablation costs ≤ 0.015). With its functional scores spread flatly across many heads, Qwen covers each function redundantly. Both families share the division of labor; the degree of redundancy differs (Section 6.5).

The observational data agree from an independent direction. Conditioning retrieval-head attention on *how* the model failed splits the errors into two families with opposite attention signatures (Figure 6.2). On *localization* failures (*off_by_row*, *wrong_col_same_row*) the retrieval heads’ attention to the target cell collapses to about a third of its correct-answer baseline, as expected if the circuit did not locate the cell. On *post-retrieval* failures (*format_mismatch*, *not_in_table*) that same attention sits *at or above* baseline (107% and 100.4%): the model attended to the right cell as strongly as when it answered correctly, then emitted a string absent from the table. What separates the two families is measured attention, so these are attention signatures consistent with the two accounts, not demonstrations of them; the split is nonetheless sharp, and the direction of each is the one its account predicts. The ordering is identical for all three re-

trieval heads and runs at 14–18 $\times$ the random-control null. Within the primary indexed condition, pooling all errors would average a bucket at 107% with one at 34% and report a spurious mid-point deficit, which is why the analysis conditions on failure type. The `format_mismatch` bucket doubles as a positive control—there the model demonstrably retrieved the right value (it produced it, in the wrong form)—and its retrieval attention duly comes out at or above baseline.

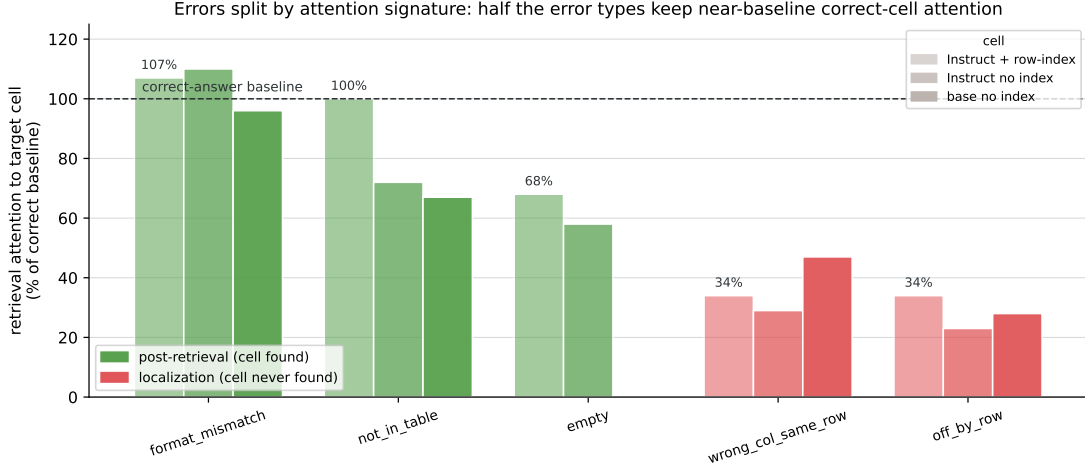


Figure 6.2: Retrieval-head attention to the target cell, by failure bucket, as a share of the correct-answer baseline. Localization failures collapse, an attention signature consistent with the cell never being located, while post-retrieval failures sit at or above baseline, a signature consistent with successful localization followed by a wrong emitted string.

6.2 Activation Patching and Marginal Repair

Denoising patching measures *marginal counterfactual sufficiency*: how much of one specific counterfactual a component can repair while the surrounding computation remains corrupt. The reverse corrupt-into-clean direction serves as a conditional-necessity check; the corruption axes, pairwise normalization, and their caveats are defined in Section 3.4. Relative to the closest concurrent table analysis, which also patches semantic row/column counterfactuals with a normalized clean/corrupt logit-difference score at residual and head-output sites [44], the present design adds cell-content and row-index-label corruptions and separates a retrieval head’s query, attention pattern, and full head output (z), so the intervention can distinguish QK routing from OV-carried content rather than localizing only a layer or head.

When a head set’s clean output is patched into the corrupt run, the matching heads that were necessary under ablation restore *little* (Figure 6.3): the four row-matching heads rescue only 0.035 of the row corruption (95% interval $[0.02, 0.05]$, bootstrapped over pairs), and the column-matching heads 0.23 of the column corruption ($[0.20, 0.27]$). The results point to a distributed matching pool in which the named heads participate but are individually substitutable, not to a four-head bottleneck. Necessity and marginal sufficiency are different estimands, and the matching stage is where they come apart.

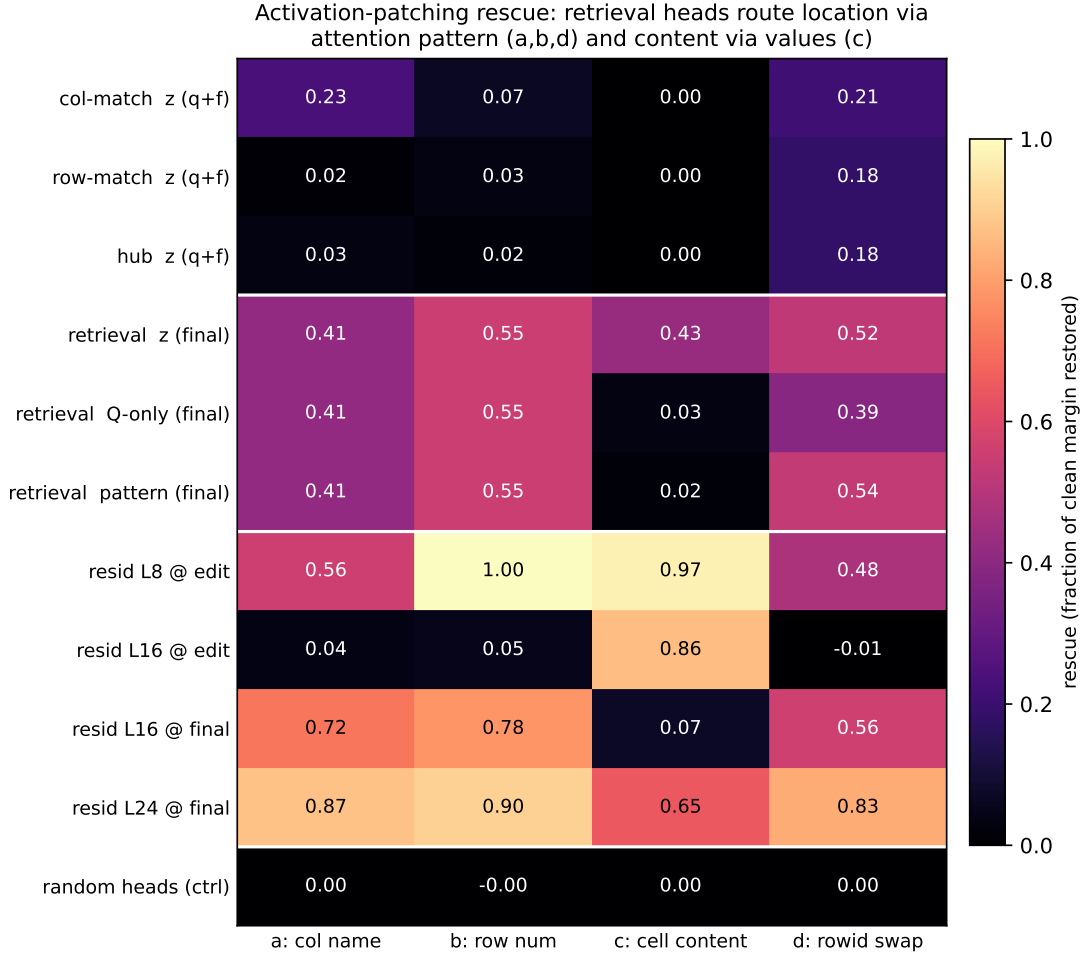


Figure 6.3: Rescued answer margin from patching each site (rows) under each corruption axis (columns), Llama Instruct, 300 aligned pairs/axis; random-head and random-position controls ≤ 0.013 . The retrieval trio supports a QK-mediated location versus post-attention content dissociation; the matching heads restore little despite their ablation necessity (Section 6.1).

The retrieval trio, by contrast, shows a causal double dissociation *inside the same three heads*. For localization corruptions (a column name, b row number, d row-index labels) patching only where the heads *look*—their attention-pattern row—rescues as much as patching their full output (0.41–0.55); for the content corruption (c) the pattern and query patches do nothing (0.02/0.03) and only the heads’ output (*z*) rescues it (0.43, [0.41, 0.45]). The result supports a QK-mediated location versus post-attention, OV-carried content dissociation. The same structure appears in where the signal lives by depth: the question-side location signal is carried at the edit position through layer 8 (rescue 0.56/1.00 on axes a/b) and has migrated to the final position by layer 16 (0.72/0.78, rising to 0.87/0.90 by layer 24), whereas cell *content* stays at the cell position until late (layer-16 edit-position rescue 0.87). The row-index manipulation makes the counting-to-matching result of Section 4.2 causal: after swapping the index labels (axis d), only 18 of 318 otherwise-valid pairs kept a positive corrupt margin. In $\approx 94\%$ of resolvable cases the model follows the *content* (the row now labeled *N*) rather than the position, with the retrieval pattern row the strongest carrier. Finally, sufficiency is bounded: the retrieval trio rescues about half the final margin, and the layer-24 residual reaches 0.83–0.90. The unrecovered portion implies contributions from other late components and interactions; these rescue values are not an additive flow decomposition. Reverse noising provides a directional consistency check: corrupting the retrieval sites in an otherwise clean run removes 0.42–0.59 of the final margin, with row-index attention-pattern noising reaching 0.59.

6.3 Joint Ablation and Direct-Logit Writers

Circuit validation uses joint ablation to test a proposed mechanism as a set rather than inferring set-level necessity by adding isolated component effects [39]. Individually the stages are partly substitutable; together they are not. Mean-ablating all twelve named heads at once collapses accuracy from 0.9100 to 0.2756, a drop of 0.634, against 0.004 for a sixteen-head random control (Figure 6.4). Because the per-stage ablations sum to only ≈ 0.22 , the joint effect is roughly $3\times$ *superadditive*: when one stage is ablated the others partially compensate, and when all are ablated together none remains to compensate. Superadditivity of this size is consistent with serial dependence between the stages, but it does not establish it: interactions can arise in a nonlinear network for reasons other than a shared serial path. The error signature is the one that column-matching ablation produces, at larger magnitude (`not_in_table` +22.9 pp, `off_by_row` +12.5 pp, `wrong_col_same_row` −27.5 pp). Qwen replicates the superadditivity: its per-set ablations are each ≤ 0.015 , yet the joint named-twelve ablation costs 0.201, against −0.001 for a larger fifteen-head random control. The pattern again fits serial dependence, at $3\times$ smaller magnitude and measured on generation accuracy rather than on the faithfulness metric of the concentration dissociation below.

The write-out, in contrast, has no detectable full-generation cost under the tested mean-ablation intervention. Direct-logit attribution (Section 3.4) nominates the writer tier; it identifies heads that write toward the answer logit, not heads whose writing is necessary. The resulting late tier is led by L31H14, but ablating the top four changes accuracy by only +0.003, and adding them to the named twelve leaves the joint effect unchanged. Qwen’s own writer tier (L24H26, L23H11, L26H26, L26H25) reproduces the null (−0.009). The write-out is redundant under

Mean-ablation necessity: the named circuit is jointly catastrophic; the DLA writer tier is not needed

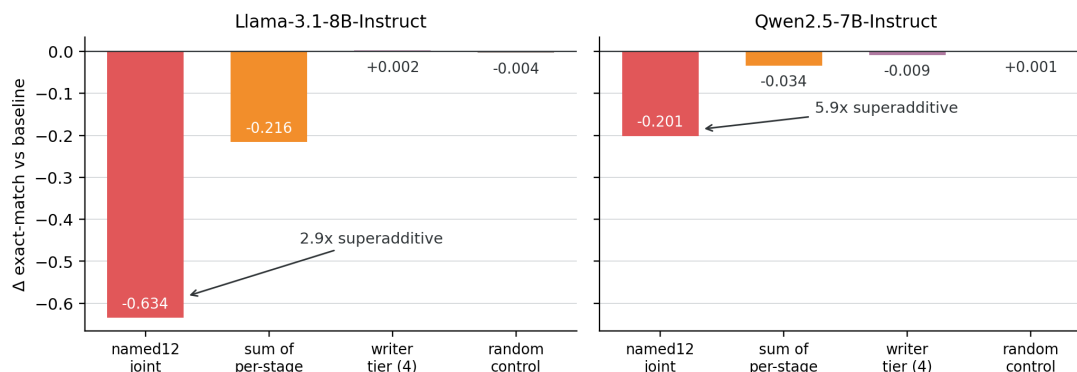


Figure 6.4: Accuracy cost of ablating the twelve named heads jointly, against the sum of the per-stage effects and a single random control draw, alongside the top-four direct-logit “writer” heads in both families.

this test.

Applying denoising to the MLP outputs asks the same counterfactual question of an alternative component class: does the head account omit a small feed-forward bottleneck? The patches never rescue more than 0.13–0.14 at the final position, with the mass spread across layers 19–31 (Figure 6.5); the large layer-0 MLP rescue looks like an extended-embedding reconstruction effect, not table computation, and random-position controls are exactly zero. No single patched MLP restores a large share of the margin, which fits a distributed or interaction-dependent MLP contribution. The account therefore carries two localized stages, not three, with a diffuse write-out tier.

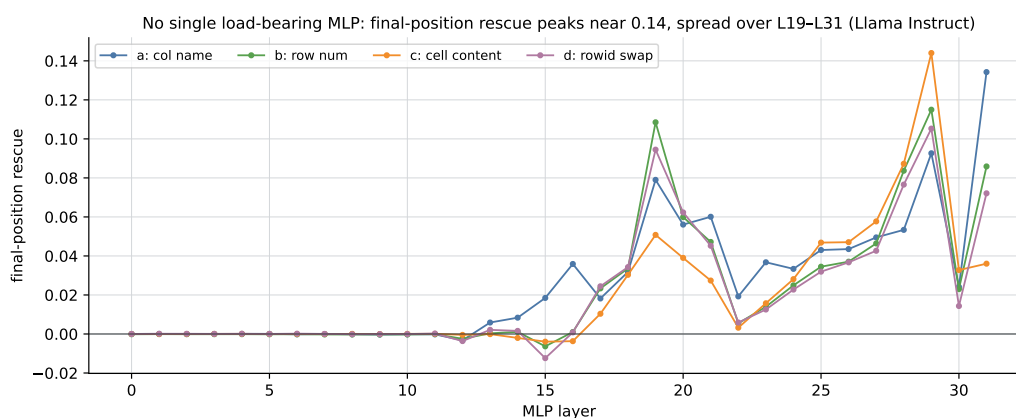


Figure 6.5: Per-layer MLP denoising rescue at the final position across the four corruption axes. The layer-0 MLP effect is consistent with reconstruction of the edited token as an extended embedding, rather than table computation; the low individual rescues are consistent with a distributed or interaction-dependent MLP contribution.

6.4 Scalable Node Attribution and Circuit Ranking

Exact activation patching is the more direct causal localizer, but an exhaustive sweep is too costly here; the head/MLP-level EAP-IG adaptation of Section 3.4 supplies the scalable node ranking instead. The score is only a screen—path-dependent, sign-discarding, and blind to redundancy—so direct patching and ablation carry the causal validation.

The node ranking agrees with the targeted patching results (Figure 6.6). The retrieval apex L17H24 is rank 1 on all four corruption axes and L16H1 ranks 2–3; the matching heads rank 49–252 by pooled attribution across the four corruption axes (out of 1024 heads), deep *even on their own axes*. Attribution estimates individual marginal contribution. The low matching-head ranks agree with the patching result in Section 6.2: the results again point toward substantial substitutability among the matching heads, while the retrieval heads carry much larger marginal signal. The two analyses draw on separately constructed samples (patching builds its pairs during the run under a 1300-token cap, attribution reads the saved 1024-token-cap bank), but they share the corpus, the construction rule, and the acceptance criteria, so the agreement is a second reading of closely related data, not an independent confirmation. The dissociation is why this thesis presents necessity and marginal sufficiency as separate axes instead of collapsing them into one ranking.

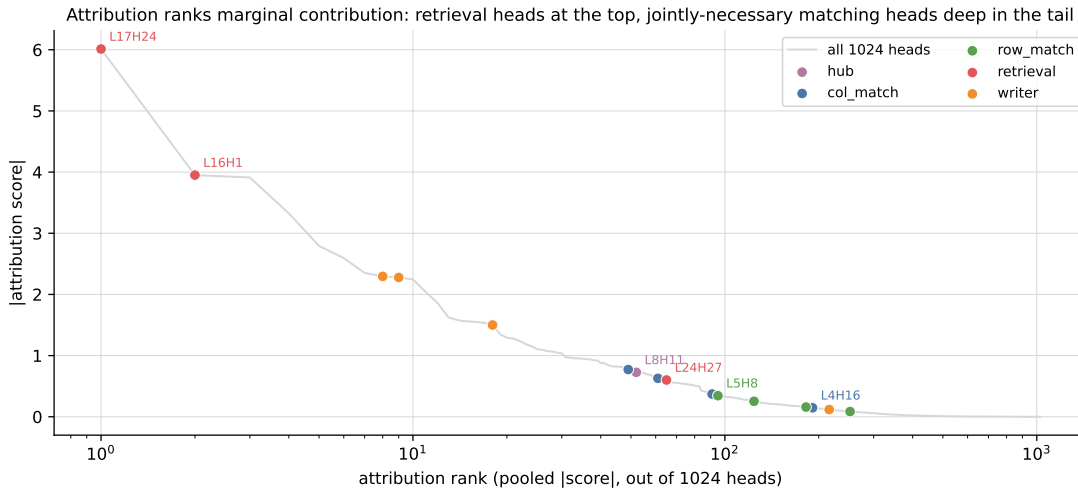


Figure 6.6: Attribution-patching rank of the named heads (out of 1024). The retrieval heads rank at the top while the matching heads rank deep even on their own axes, a pattern consistent with substantial substitutability despite their joint necessity (Section 6.3).

6.5 Faithfulness and Circuit Adequacy

A ranked list of individually interesting heads is not yet an adequate circuit. Adapting the adequacy criteria that Section 2.3 traces from the IOI study and MIB, the top- K sweep measures faithfulness; the inverse-complement experiment asks whether the unpatched complement can

compensate for the selected core rather than measuring IOI completeness; and tier drops provide only partial, tier-level minimality.

The replacement is the counterfactual interchange intervention of Section 3.4: the clean token sequence is run while every non-circuit head output is overwritten with its cached value from that pair’s corrupt run, and faithfulness $f(C_K)$ is the pair-normalized recovery of the first-token margin, averaged over pairs. Top- K curves replace an intractable search over all subsets, so they measure the efficiency of a declared ranking rather than proving that no smaller circuit exists. The all-head endpoint satisfies $f(\text{all}) = 1$ by construction, while the zero-head endpoint is $f(0) = 0.000$ empirically. At $K = 0$, clean embeddings and unpatched, recomputed MLPs are insufficient to recover the pairwise margin when every head output is replaced by its corrupt-run value. This supports a heads-only circuit class for the scored intervention, but does not isolate direct head mediation from downstream effects of those head replacements on the MLPs.

6.5.1 Faithfulness Against Circuit Size

An $f = 0.8$ circuit requires hundreds of heads; the sixteen named and writer heads do not approach it (Figure 6.7). How sharply that number is resolved differs by ranking and family. On the coarse K grid (Section 3.4) the interpolated crossing is 278 heads in Llama (27% of attention heads) and 181 in Qwen (23%) under the pure attribution ranking, and 274 and 180 under the named-first ranking. Those interpolations span the wide grid gaps $205 \rightarrow 512$ and $102 \rightarrow 205$, across which the curve rises faster than linearly, so they locate the crossing only to within a grid interval. Direct evaluations run for the minimality experiment below bracket the named-first crossing more tightly, with pair-level 95% intervals in brackets: $f(219) = 0.754$ ($[0.73, 0.77]$) and $f(274) = 0.893$ ($[0.88, 0.91]$) in Llama, $f(144) = 0.794$ ($[0.78, 0.81]$) and $f(180) = 0.852$ ($[0.84, 0.86]$) in Qwen; these intervals come from the sweep and are over pairs, not clustered by source table (Section 9.3). Interpolating inside those brackets puts the crossing near 237 heads (23%) in Llama and 148 (19%) in Qwen. The Llama bracket is resolved at both ends, while Qwen’s is not: its interval at $K = 144$ already contains 0.8, so the Qwen crossing may sit at or below 144 and 148 is not a precisely located boundary. The attribution-ranking crossings have no such direct brackets at all and remain coarse-grid estimates. What holds across grids and rankings is the scale—roughly a fifth to a quarter of the heads—not any particular K .

Because the pooled estimand gives each corruption axis exactly one quarter of the weight, the axis-specific curves provide a weighting-sensitivity check. On the named-first ranking, coarse-grid linear interpolation places the $f = 0.8$ crossings for column-name, row-number, cell-content, and row-index-swap axes at approximately 405/199/158/210 heads in Llama and 196/177/100/180 in Qwen. The macro-average circuit size would therefore change under another axis distribution; these interpolations summarize the saved grid and do not locate precise thresholds.

At $K = 12$, the named-first circuits give $f = 0.063$ in Llama and $f = 0.0003$ in Qwen; at $K = 16$ they reach 0.14 and 0.06. They are not by themselves a faithful circuit. In Llama the named heads nevertheless form a dense per-head core over part of the range: named-first beats the pure attribution ranking at $K = 16, 26, 51$, and 102, having lost to it at every nonzero smaller size ($K = 1, 2, 5, 12$), where attribution’s single best-scoring heads recover more (Appendix C). Random rankings require 78–88% of Llama’s heads and 89–93% of Qwen’s for the same $f = 0.8$

Counterfactual-interchange faithfulness vs circuit size

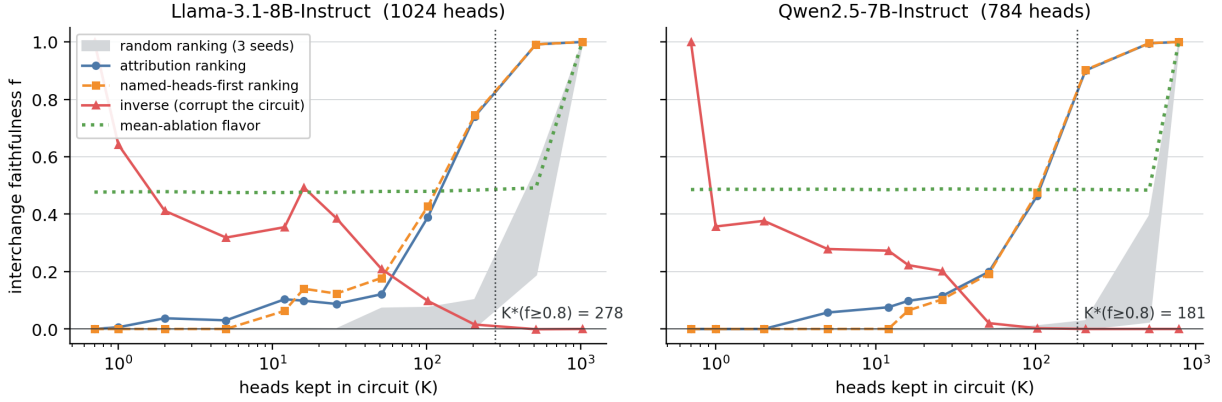


Figure 6.7: Counterfactual-interchange faithfulness against circuit size K for both families, with attribution, named-first, and random rankings (three random seeds, drawn as a range, not a confidence band). The all-head endpoint is 1 by construction and the zero-head endpoint is 0 empirically. Under the same hybrid intervention, the inverse trace overwrites the top- K rather than its complement; it is the separate quantity f_{inv} of the inverse-complement test below: it runs *downward* from 1 and is not read on the same footing as f .

threshold. Small- K rescue is content-first: at $K = 12\text{--}16$ essentially all rescue is the content axis (carried by the retrieval trio and writer), while the localization axes stay ≈ 0 until $K > 51$. Content recovery concentrates in a few heads; localization recovery requires many. Appendix C tabulates the complete numeric grids behind these curves, with bootstrap intervals and axis-resolved values.

Combining the faithfulness curves with the ablation necessity of Sections 6.1–6.3 gives the cross-family contrast of Figure 6.8, with the caveat that the two halves are different estimands: necessity here is a full-generation accuracy cost under mean ablation, sufficiency a first-token interchange margin on selected pairs, so the families are compared on two measurements rather than on one quantity that reverses. Llama concentrates necessity (the named twelve cost 0.634; column matching 0.136) while spreading sufficiency ($K^*(0.9) \approx 400$ heads, 39%). Qwen shows the opposite pattern on each measurement taken separately: diffuse necessity across the tested small, functionally selected sets, but sharper sufficiency ($K^*(0.9) \approx 205$, 26%; at $K = 205$ Qwen reaches $f = 0.90$ against Llama’s 0.74). Both $K^*(0.9)$ figures are coarse-grid interpolations and both hold under either ranking (400 attribution against 398 named-first in Llama, 205 against 204 in Qwen), so the dissociation is a property of the families rather than of the ranking used to order their heads. This difference, not raw accuracy, is the quantity Chapter 7 connects to the families’ training.

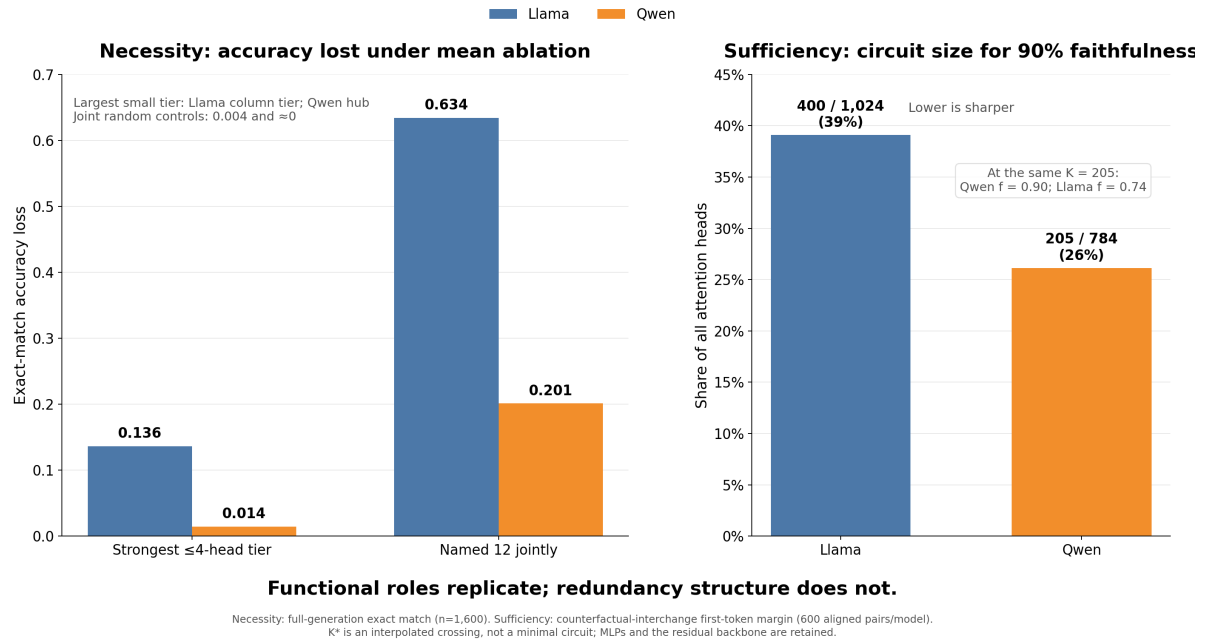


Figure 6.8: The two halves of the cross-family contrast, measured on different quantities. Left: full-generation accuracy lost under mean ablation ($n = 1,600$) for the strongest small tier and for the named twelve jointly; the joint effect is large in both families while the strongest small tier differs (Llama’s column tier, Qwen’s hub). Right: the share of heads needed for 90% interchange faithfulness (39% of Llama’s 1,024 against 26% of Qwen’s 784). Functional roles replicate across the families; the redundancy structure does not.

6.5.2 Inverse-Complement Necessity

The next adequacy question is whether an unpatched complement can compensate when a ranked core is overwritten. Under the same hybrid setup, inverse-complement necessity is tested on the attribution ranking by replacing the top- K rather than the non-circuit head outputs. The resulting score falls to $f_{\text{inv}} = 0.016$ for Llama at $K = 205$ and 0.003 for Qwen at $K = 102$. Thus the unpatched complement cannot carry the margin under these interventions. These attribution-ranked core sizes differ from the named-first evaluation circuits used for the tier drops below. Unlike the stronger IOI criterion, which compares circuit and full-model responses under matched knock-outs, this test establishes necessity of the specified cores under the hybrid intervention, not that they account for every relevant outside component. In particular, it cannot detect complement components that matter only in concert with an intact core. Corrupting a single head, L17H24, removes 36% of the margin in Llama (64% for Qwen’s apex) under this interchange intervention.

6.5.3 Tier-Level Minimality

Minimality is tested only at the level of *tiers*: whether each *named* tier, and the circuit’s low-ranked tail, can be removed without cost. Tiers are dropped from named-first *evaluation circuits* at $K = 274$ (Llama) and $K = 180$ (Qwen), sizes fixed from the coarse-grid crossing estimates before the direct evaluations above bracketed the crossings more tightly, so they are evaluation points, not $f = 0.8$ minima. Direct evaluation at those sizes gave $f = 0.893$ and 0.852 , respectively (Figure 6.9). Removing the three retrieval heads costs 0.578 (Llama) / 0.332 (Qwen), substantially more than either of the two random in-circuit reference drops: the retrieval trio is the least replaceable tier in both families and has the largest tier-drop effect. Evidence for the other tiers is mixed. In Llama the writer (-0.216) and column-matching (-0.178) tiers exceed the random-drop reference range, while hub and row matching do *not*, so this test does not establish them as minimal; in Qwen only the late stages (retrieval -0.332 , writer -0.265) are concentrated. The bottom-20% drop still costs $0.139/0.059$ (these drops are the direct $f(219)$ and $f(144)$ evaluations quoted above), showing that this tail collectively carries signal, not that every tail head is necessary.

Two limits keep this from being a minimality result in the strong sense. A full minimality claim would show that no proper subset of the ≈ 200 -head circuit reaches the same faithfulness, a component-wise test this thesis does not run. And the reference the tier drops are read against is the range of *two* random in-circuit drops, far too small a sample to be a null distribution (Section 9.3.3), so “exceeds the random range” is a coarse comparison, not a test. Since the hub and row tiers fall inside that range, this thesis claims partial, tier-level minimality evidence at the retrieval core, not a minimal circuit.

Minimality: dropping a named tier from the faithful circuit — the retrieval trio is the core in both families

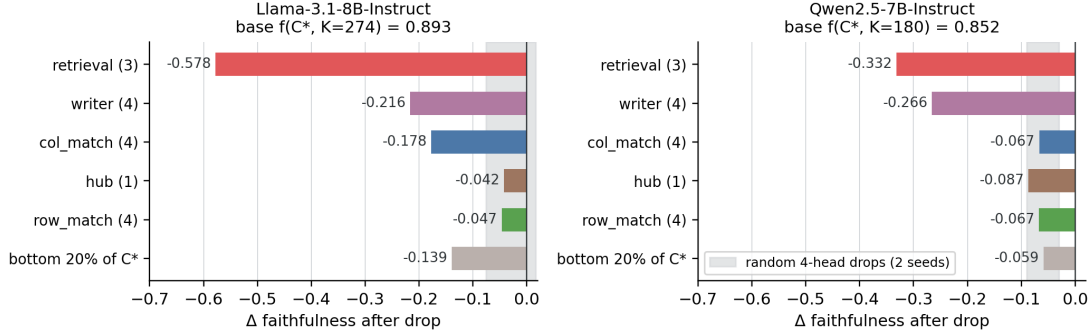


Figure 6.9: Faithfulness drop from removing each named tier from the named-first evaluation circuit, against the range of two size-matched random in-circuit drops (a reference range from two draws, not a null distribution). The retrieval trio is the least replaceable tier in both families; the hub and row tiers fall inside the random range.

6.5.4 Dependence on Metric and Ablation Flavor

Two qualifications apply to these numbers. First, the writer tier *dissociates by metric* in both families: it has a substantial tier-drop effect on the first-token margin within the fixed interchange circuit ($-0.216/-0.265$) yet has no detectable full-generation cost under mean ablation ($+0.003/-0.009$, Section 6.3). Other late components plausibly compensate during generation. The measurements answer different questions. Second, the results depend strongly on the ablation flavor and the metric. The same circuits scored under interchange $\times$ margin give the curves above; under pooled-mean $\times$ margin f is a flat ≈ 0.48 from $K = 0$ to $K = 512$ (a mean-ablated model has ≈ 0 margin regardless of what is kept); and under pooled-mean $\times$ generation f is 0.000 at every $K < \text{all}$. The choice of flavor therefore determines whether the measurement is informative, not merely its value [25]. Every faithfulness figure in this thesis is labeled with its flavor and metric, with interchange $\times$ margin as primary: it follows MIB’s counterfactual-ablation protocol and, because the pairs are minimal edits, asks which selected heads can collectively reconstruct the clean-versus-corrupt margin under this intervention.

Chapter 7

Alternative Explanations for Format and Model Differences

This chapter tests alternative explanations for the format and family differences: a delimiter-counting account of column location, three measured tokenizer properties, and the families’ documented training-data differences, the last held as hypothesis. It compares the results with the delimiter-counting account of cell location of Zhang et al. [44], and a further experiment tests whether the measured column match is semantic or lexical.

7.1 The Pipe-Deletion Control

Zhang et al. [44] propose, on synthetic pipe-delimited tables, that models localize columns by *counting pipe delimiters*, a RoPE-based ordinal mechanism. Under that mechanism, deleting the pipes should destroy column matching here. Re-extracting the QK selection margins of the named heads (Section 6.2’s machinery: the attention-logit margin of the correct key over matched distractors) on a *whitespace render*—the identical real tables with every pipe deleted and columns space-aligned ($n = 977$)—leaves the matching heads essentially unchanged (Figure 7.1). Eleven of the twelve named heads retain 95–122% of their piped margins: on the column axis L4H16 is at 100%, L14H19 at 99%, L10H2 at 116%, and the hub L8H11 at 101% (only L8H1 drops, to 39%); of the four row-matching heads L5H8/9/11 sit at 115–122% and L14H17 at 95%, and the retrieval trio holds 101–103%. The ranked head identities remain substantially overlapping (top-eight overlap 5/8, 7/8, 7/8 across the three maps), and the behavioral cost of removing the pipes is only 0.039 ($0.912 \rightarrow 0.873$). The same heads therefore compute near-identical selection margins once the pipe delimiters are gone. What the whitespace render removes is the pipes specifically: the newline that ends each row and the space alignment that keeps columns visually registered both remain, so the test rules out *pipe* counting by the named route rather than every positional or whitespace cue. Within that scope, column matching anchors on cell *content*, not on pipe tokens. Delimiter counting is at most an auxiliary route here: that behavioral cost, together with the one head that does drop (L8H1), bounds its residual contribution rather than eliminating it. This is less a contradiction of Zhang et al. [44] than a difference of regime, since their synthetic, semantics-free tables may genuinely force positional counting.

On the shared task and model, however, the measured cue is cell content, not delimiters.

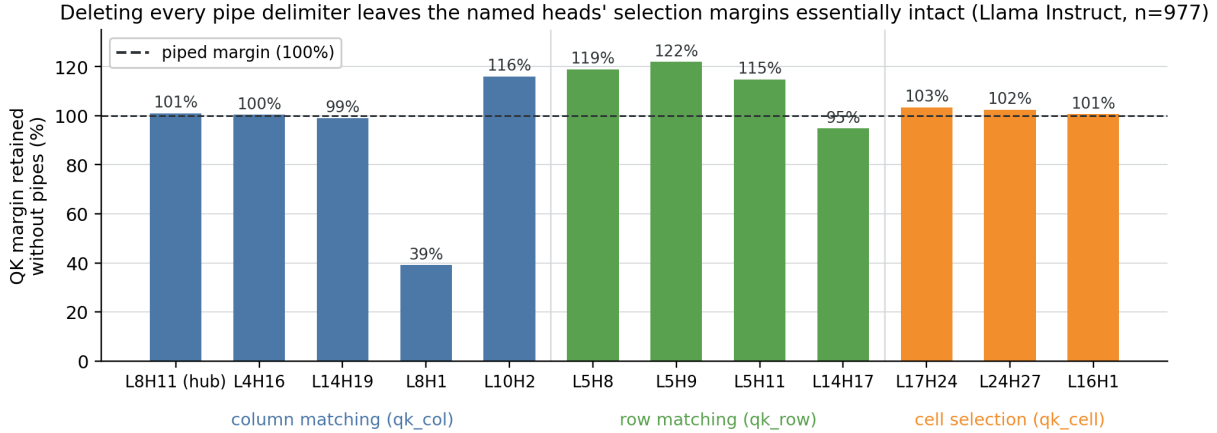


Figure 7.1: QK selection margins of the named heads on the piped render versus the pipe-free whitespace render (identical real tables, $n = 977$). Eleven of twelve named heads retain 95–122% of their margins; top-eight overlap is $5/8$ for column matching and $7/8$ for each of row matching and retrieval.

7.2 Lexical versus Semantic Matching

The pipe-deletion test shows matching is *content*-based, not delimiter-based, but it leaves a further question open: is that content match *semantic* (the model understands that `Score` and `points` denote the same column) or merely *lexical* (it copies the exact header string)? A paired experiment separates the two. On the same mappable-question subset, the column word inside the question is replaced with a near-synonym while the table header is left exact (asking for ‘points’ when the header reads `Score`), and the QK margin is measured toward the geometrically identified target header, so it registers whether the synonym query still selects the correct column. Under the substitution the four dedicated column-matching heads fall to a mean of -4% of their margin (L4H16 -7% , L14H19 -6% , L8H1 -4% , L10H2 2% ; Figure 7.2): the margin toward the correct header is not merely reduced but eliminated, and on average slightly negative, meaning the distractor columns are selected about as strongly as the target. The one partial exception is the hub L8H11, the most robust matcher, which still holds 47% ; folding it in, the five-head aggregate averages 6% . The positive controls indicate the effect is specific to the swapped column word: the row-matching heads retain $\approx 100\%$ and the retrieval trio a mean 92% (its members $101\%/95\%/79\%$), since neither depends on the column word.

These are aggregate margins. The run stored per-head means over the paired questions rather than the per-question or per-table values, so no interval attaches to any percentage above and no per-lexeme breakdown is available: it is not measured whether the collapse is uniform across the synonym map or driven by a subset of substitutions. The tables were also taken as a deterministic sorted prefix of the corpus rather than sampled (Section 3.5). The effect is large enough that

sampling noise is an unlikely explanation of its sign, but its magnitude is a point estimate on one constructed substitution set.

The named QK route is not robust to these near-synonym substitutions: within that route the match behaves like an exact-string, induction-like match on the header token [29], not a conceptual one. The two experiments agree with a content-based account: the match survives pipe deletion and fails under a header-word substitution, so the content it keys on is the header string itself. This is a property of the measured route, not of the model’s semantic capacity: the measurement covers selection margins rather than end-to-end behavior, so downstream compensation by another pathway would go unseen; it covers the four named column-matching heads and the hub, not the remaining thousand heads; and it uses one constructed set of 363 near-synonym substitutions, which samples the neighborhood of the header string rather than the space of paraphrases. The question template is also held fixed: it asks for the synonym as though it were the header (*the value in the ‘points’ column*). A phrasing that marks the substitution as approximate (asking for *the column that measures points*) poses a different task, closer to paraphrase resolution than to the lexical match measured here, and whether it would re-engage this route or recruit another is untested.

A post-hoc behavioral split offers limited convergent evidence. At widths 5–7 and 8–10, tables whose header row is at least half numeric-like or blank trail majority-lexical-header tables by 0.232 and 0.265, respectively (Section 4.4). The difference is only 0.026 in the sparse ≥ 11 bin, and the numeric/empty groups comprise just 21 underlying tables in total. This association is compatible with lexical matching being easier when headers provide distinctive strings, but it neither measures the QK route on those tables nor separates header form from source-template difficulty.

7.3 Tokenization Controls

The most concrete low-level explanation for the family gap—that Llama and Qwen tokenize tables differently enough to matter—can be tested against three direct measurements (Table 7.1), and none of them supports it. Across all 1,340 tables, pipe handling is byte-identical: both tokenizers reduce the corpus to the same three pipe pieces with identical counts, and *cell/token boundary alignment is exactly 1.000 for both*, so neither model pays a misalignment penalty. The tokenizers do differ in digit granularity: 1234 becomes [123, 4] for Llama but [1, 2, 3, 4] for Qwen, costing Qwen $\approx 11\%$ more tokens per cell (5.53 vs 4.97). Yet Qwen performs *better*, so token efficiency does not explain the gap either; if anything it runs the wrong way.

What this rules out is narrower than “the tokenizer.” The three quantities measured here—delimiter tokenization, cell boundary alignment, and token efficiency—are the tokenizer properties with an obvious mechanism by which they could produce the gap, and all three come out null or backwards. Other ways the two vocabularies might differ are untouched: segmentation of the specific header and cell strings, subword frequency effects on the lexical matching route of Section 7.2, and interactions between segmentation and position, and separating those properly would need an intervention (retraining or transplanting a tokenizer) that is out of reach here. The claim carried forward is the measured one: no simple tokenization difference accounts for the cross-family gap, which removes the most readily available low-level confound without closing

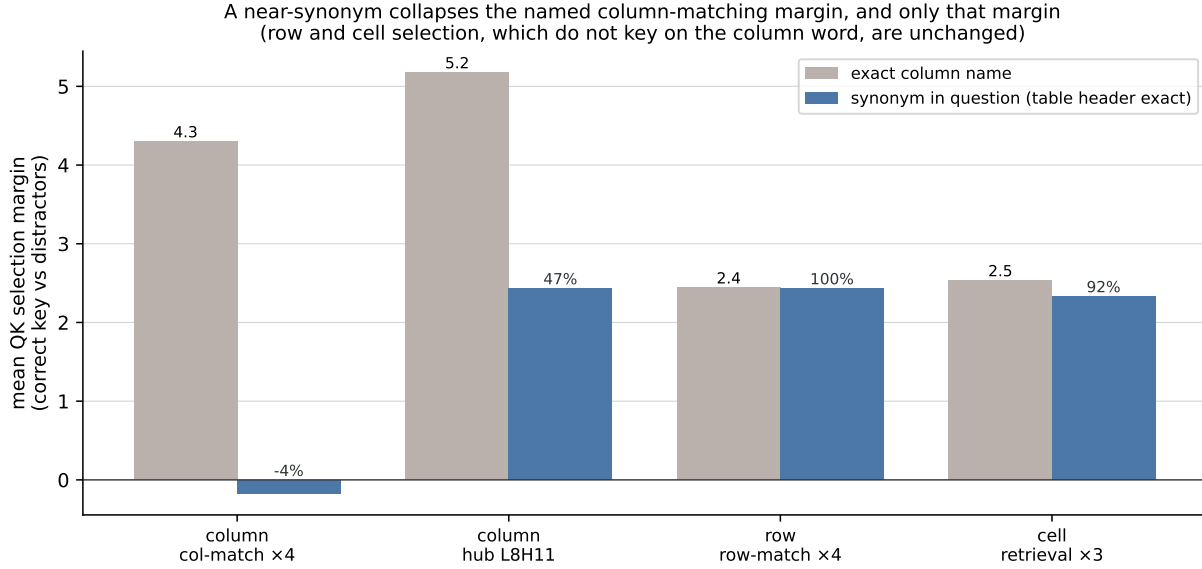


Figure 7.2: Mean QK selection margin for exact column names versus a question-side synonym (table header left exact; $n = 363$ paired questions). The four dedicated column-matching heads collapse under a synonym while the row-matching and retrieval controls are largely preserved. This bounds the measured route, not the model’s semantic capacity.

tokenizer property	Llama-3.1-8B	Qwen-2.5-7B
pipe pieces (identity, counts)	identical	identical
cell/token boundary alignment	1.000	1.000
digit granularity (1234)	[123, 4]	[1, 2, 3, 4]
tokens per cell	4.97	5.53 (+11%)
base 0-shot markdown accuracy	0.439	0.580

Table 7.1: Tokenizer statistics over all 1,340 tables. None of the three simple tokenization differences explains the cross-family gap; finer segmentation effects are not measured here.

the question.

7.4 A Training-Data Hypothesis

A training-data account remains; I hold it explicitly as hypothesis rather than causal claim, because both corpora are closed: these are documented-fact correlates anchored to the measured mechanistic quantities, not controlled experiments. Two correlates are consistent with the evidence. First, **markdown exposure**: Llama 3’s pipeline reports *removing* markdown markers from web data (“we find markdown is harmful... we remove all markdown markers”) [23], whereas Qwen2.5 reports no such step and explicitly targets structured input, “tables and JSON” [34], consistent with Qwen base’s large lead on plain markdown (0.580 vs 0.439) and with markdown remaining easier than CSV in both families. Second, **code and structured-data emphasis**:

Llama 3 reports 17% code and 25% mathematical and reasoning tokens within its 15.6T-token corpus [23], while Qwen2.5 describes its 18T-token mix as emphasizing code and math and as incorporating data from its specialist Coder and Math pipelines [12, 34]. The public Qwen reports do not state what fraction of the general model’s mix those corpora constitute, so any claim of heavier structured-data exposure is a qualitative reading of the documentation, not a measured ratio. Code pretraining has been shown to improve entity and state tracking [17] and structured-task performance [33]. A mechanism of this kind could produce the concentration dissociation of Section 6.5 (many parallel table-reading pathways, each individually stronger), but nothing measured here tests it.

Instruction tuning is excluded as the primary factor by this thesis’s own measurements: at byte-identical prompts, tuning moves behavior by only +0.014, and it re-weights what heads write more than what they select. Ranking every head by its QK selection scores under the base and the tuned weights gives Spearman correlations of 0.94–0.99 between the two models, against a within-model odd/even split-half reliability ceiling of ≈ 0.99 ; that is, tuning barely moves the selection maps relative to their own measurement noise. The direct-logit ranking agrees less well (0.89 against the same ceiling). Tuning therefore cannot account for the *base*-model gap. The quantitative signatures any provenance account must explain are therefore the K^* sufficiency dissociation and the necessity contrast of Chapter 6, not raw accuracy, with the tokenizer ruled out in Section 7.3. Taken together with the real-table data regime, the causal row-index mechanism, the cross-family concentration analysis, and the direct delimiter test of Section 7.1, this work differs from Zhang et al. [44] in four respects: it uses real tables where theirs are synthetic; it finds content, not delimiter position, to be the measured cue (tested head-to-head); it gives a causal account of the row-index mechanism; and its concentration analysis is cross-family.

Chapter 8

Formatting Guidance

Behavioral work has shown that table formatting moves accuracy without explaining why [22, 35]. The preceding chapters name the operations the model performs, allowing each recommendation to be tied to a measured effect and an explicit limitation. The scope is this thesis’s task: short flat tables and single-cell lookup. Each item states the recommendation, the operation it serves, and where the evidence for it sits.

8.1 Recommendations the Mechanism Supports

Add an explicit row-index column. This changes the addressing problem from counting rendered rows to matching an explicit key; it is not a cosmetic formatting tweak. The change raises accuracy by 0.16–0.33 on markdown and 0.18–0.28 on CSV, in every cell of the condition grid and in both families (Section 4.2), because the non-indexed condition has no explicit content-addressable landmark corresponding to the requested row number. Routing from a generic “row” keyword to the row header is weak; what the matching stage binds is the requested *index value* against the contents of the index column (Section 5.3), and ablating the heads that perform that binding selectively lands the model on the wrong row (Section 6.1). The index column supplies the only row landmark measured by the localized row-matching route. The non-indexed depth profile establishes only that retrieval follows the available column-matching signal; it does not measure row-index routing without an index (Section 5.1). The index also blunts the widest-bin loss, especially on markdown: it cuts that loss by roughly 60%, against 26% on CSV (Section 5.6).

Use an explicit row key rather than relying on row position. The benefit of the index survives rendering the rows in reverse, so that the labels contradict physical order (Section 4.2), and swapping the labels causally confirms it: in $\approx 94\%$ of resolvable cases the model follows the row now *labeled* N rather than the row in position N (Section 6.2). I tested only numeric indices, in forward and reversed order; the permissive extrapolations this result invites (non-contiguous, unsorted, filtered, or paginated identifiers) are stated as hypotheses in Section 8.2. One caveat: under reversed order, two-shot prompting scores *below* zero-shot in both families, by 0.038–0.061, while the same examples move the ordinary indexed condition by -0.010 to

+0.057, without the larger reversal-specific penalty, so demonstrations should not be expected to compensate for a reversed table. The design that isolates this interaction and its open cause are set out in Section 4.2.

Prefer markdown to CSV, but not for its delimiters. Markdown beats CSV in all twelve matched Llama cells, by 0.06–0.23 (Section 4.1). The pipes are not what the advantage is made of: deleting every one of them and space-aligning the columns leaves eleven of the twelve named heads at 95–122% of their piped margins and costs 0.039 (Section 7.1), and no simple tokenization difference accounts for the family gap either (Section 7.3). The localized column-matching route keys on cell *content*. The pipes are therefore not necessary for that dominant localized route, though not irrelevant: one named head falls to 39% of its piped margin and the deletion still costs 0.039 behaviorally. The reason to prefer markdown is the measured margin, not any property of its punctuation. A space-aligned render is not CSV, so these two manipulations bound the delimiters’ contribution rather than decomposing the format gap.

Row-index placement is usually flexible. Rows are anchored by index content rather than by its ordinal location, so moving the *row-index* column leaves the ranked routing maps essentially unchanged and costs only 0.02–0.06 in accuracy across the main grid (Section 4.3). This supports using an existing key column where it already occurs rather than moving it mechanically to the left. The claim is specific to the row landmark, not to the queried data field. It also weakens in the pooled widest-table subset, where first-column placement gives a larger mitigation than randomized placement (Section 5.6).

8.2 Provisional Recommendations

Treat non-standard row keys as a hypothesis, not a result. Because the causal evidence shows binding to index *content* (Section 6.2), the result motivates testing whether identifiers that are non-contiguous, unsorted, or not one-based (including a filtered or paginated view that keeps the source table’s identifiers) work as well, provided the string the question asks for appears in the column. It is a hypothesis about what should generalize, not a tested condition: the corpus varies only numeric indices in forward and reversed order, and row lookup keyed by a name or a date is explicitly outside the evaluated conditions (Section 3.2).

Keep frequently queried fields early in wider tables. Target position is nearly free through about eight data columns, but at exact widths 9–13 the last three columns trail the first three by 0.063–0.145 (Section 4.4). The broader wide-table association also reaches the first three columns, so reordering would address only the additional right-edge component if this result transfers. This is observational advice: column semantics are not randomized, and later columns may be systematically different from earlier ones.

Project or split very wide tables, but do not treat ten columns as a cutoff. The pooled ≥ 11 bin is difficult across the grid, and the measured matching and retrieval signals decline with it

(Section 5.6). Exact-width behavior is more qualified: lexical-class accuracy is broadly flat from widths 4 through 11, widths 12 and 13 contain only two and five tables, and the width-14 cliff rests on 15 mostly related monthly tables (Section 4.4). Projection is therefore a prudent way to avoid an empirically difficult tail, not a rule supported by a universal breakpoint. The mechanism and the width at which the advice transfers to a new corpus remain open.

Use the header’s wording, and prefer distinctive lexical labels. Replacing the column word in the question with a near-synonym, leaving the header exact, collapses the four dedicated column-matching heads to a mean of -4% of their selection margin while row-matching and retrieval controls hold (Section 7.2). Separately, tables with at least half numeric-like or blank headers trail lexical-class tables by 0.232–0.265 at widths 5–10, but not in the ≥ 11 bin (Section 4.4). Together these results favor using the exact, distinctive vocabulary that questions will name. Neither establishes an end-to-end causal effect: the synonym test measures five heads on 363 paired questions, while the header split is post hoc, combines numeric-like with blank headers, and contains only 21 such underlying tables across the reported bins.

8.3 Limits of Formatting Interventions

The mechanism-backed formatting evidence acts most directly on one of two error families. Conditioning retrieval attention on how the model failed splits the errors cleanly: on localization failures the retrieval heads’ attention to the target cell collapses to about a third of its correct-answer baseline, whereas on post-retrieval failures it sits at or above baseline (Section 6.1). In the second family the retrieval heads attended to the target cell at full strength and the emitted string was still wrong, an attention signature consistent with successful localization, not proof of it. The row-index intervention and the localized QK tests address the first family; the post-hoc width, target-position, and header-class associations were not decomposed by failure family, so their pathway remains unresolved. The residue left after applying the row-index condition shifts partly into the second family: `not_in_table` grows alongside `wrong_col_same_row`, so what remains is a mix of post-retrieval failure and a column-side localization failure that the row index does not address (Section 4.2). For the post-retrieval share the natural remedies are on the output side: constrained decoding, a stated answer format, or tolerant matching. The experiments here do not show that the tested layout interventions address this error family.

Two features of the account suggest the guidance is not specific to one model. The recommendations name functions, not coordinates, and the division of labor into column matching, row-index matching, and retrieval replicates in both families even though the head positions do not (Sections 5.3, 6.1). The two items with the largest measured effects, the row-index column and the format choice, were measured behaviorally in both families and serializations; the post-hoc geometry associations also pool both families, although over the same table corpus. The detailed causal and QK evidence for content-addressed keys and exact header wording is Llama’s. Transfer of those mechanisms is inferred from the shared division of labor rather than demonstrated.

Chapter 9

Conclusion and Future Work

9.1 Summary

A decoder-only language model receives a serialized table as a flat token sequence, but answering a cell-lookup question requires recovering both a row and a column. This thesis separated that puzzle into three linked questions: which input cues make the address recoverable and which errors they repair; whether lookup decomposes into identifiable internal operations; and whether causal interventions support that decomposition and distinguish simpler explanations for format effects. I answered them with a 48-condition behavioral and attention grid spanning two model families, followed by ablation, activation patching, and circuit-adequacy tests in a controlled causal regime.

The strongest behavioral clue is the explicit row-index column. Adding one raises accuracy by 0.16–0.33 on markdown and 0.18–0.28 on CSV in every cell of the grid, chiefly by repairing answers from the correct column but the wrong row. Moving the index column costs little, its benefit largely survives row-label reversal in zero-shot prompts, and the aligned-pair label-swap test usually makes the model follow the row carrying the requested value rather than the row occupying that ordinal position. The index therefore does more than add a helpful token: it changes implicit row counting into matching against a content-addressable landmark (Sections 4.2, 4.3, and 6.2).

Following that landmark through the network produces a two-stage account. Attention profiles place column-name matching before correct-cell retrieval in all 48 completed conditions; where an index exists, a separate route binds the requested index value to its table occurrence. In the primary Llama regime, row-tier ablation selectively increases wrong-row failures, while patching the retrieval heads supports a dissociation between QK-mediated location and post-attention, OV-carried content. The behavioral and causal answers thus fit together: an earlier *matching* stage establishes row and column landmarks, and a later *retrieval* stage uses that address to select the cell (Sections 5.1, 5.3, and 6.2).

This is a claim about functional organization, not a complete twelve-head circuit. Joint mean ablation of the named Llama heads reduces accuracy by 0.634, compared with 0.004 for one larger random control, so the set contains causally important components. Under the heads-only interchange metric, however, recovering $f = 0.8$ requires roughly 19–27% of attention heads,

depending on family and ranking. The named heads are therefore a causal core within a broader distributed implementation. Llama and Qwen share the matching–retrieval division while differing in head coordinates and in where necessity and sufficiency concentrate (Sections 6.3 and 6.5).

The alternative-explanation tests complete the story. Markdown outperforms CSV in all twelve matched Llama conditions, yet deleting every pipe preserves most named-head selection margins, although it still costs accuracy and weakens one head. Replacing the queried column name with a near-synonym instead collapses the dedicated column-matching heads’ aggregate margin. The localized route therefore depends primarily on exact, nameable content rather than pipe counting, without implying that the model as a whole lacks semantic alternatives (Sections 7.1 and 7.2). Taken together, the evidence favors a concrete addressing story within the tested task: numeric row indices and exact column names provide landmarks, matching binds the question to them, and later retrieval uses the resulting address. This supports row-index columns and markdown within the tested task; recommendations about other key types, table geometry, and header classes remain provisional.

9.2 Interpretation and Scope

The findings above operate at behavioral, functional, and head-specific levels, and those levels do not generalize equally. The main interpretive choice is how the controlled row-index condition relates to ordinary tables. Tables in the wild identify their rows by something written in the row—a name, an identifier, a date—and questions about them refer to rows the same way. A query that can reach row 7 only by counting seven row boundaries is the unusual case, and it is included because removing the row landmark isolates what the matching stage contributes. The common cross-condition metric shows only that retrieval follows the available column-matching signal without an index; it does not show a failed row-matching stage there (Section 5.1). The stronger row-binding claim is restricted to indexed conditions, where the relevant landmark exists.

The evidence supports generalizing operations, not head coordinates. Binding to the index value motivates the hypothesis that other nameable keys may also work, which is why Section 8.2 treats non-contiguous, filtered, and paginated identifiers as tests to run rather than results. The preference for markdown does *not* transfer through its delimiters, because the pipes turn out not to be what the column route reads (Section 7.1). And the recommendation to use the header’s exact wording is bounded by the same mechanism that supports it: an induction-like lexical match can weaken under paraphrase. The tested near-synonym weakens the named route, so robustness to paraphrase cannot be assumed.

The quantitative results are the most setting-specific. The circuit sizes are conditional on a declared ranking, ablation flavor, metric, and heads-only circuit class; the same sets produce qualitatively different profiles under other replacements and metrics (Section 6.5.4). Head coordinates do not transfer across families, and neither does the redundancy structure: Llama and Qwen share the division of labor while concentrating necessity and sufficiency in opposite places (Section 6.5). On this evidence, the division of labor is what recurs across families, while the head-level implementation is family-specific.

Beyond tables, the matching stage resembles an induction-like operation: locate a landmark in context, bind a query token to it, and use that binding to drive a later prediction [29]. Whether

the same structure appears in other sequences with latent organization—JSON objects, source code, or game transcripts [19]—is untested; this thesis provides evidence in one setting where the addressing step can be isolated and scored exactly.

9.3 Limitations

The individual chapters state their own caveats where the measurements are made. Collected in one place, the constraints on what this thesis establishes fall into three groups: external validity, sample and metric provenance, and the strength of the uncertainty and controls.

9.3.1 External Validity

Two families, short flat tables, numeric keys. Everything here is measured on two open-weight models of seven to eight billion parameters, on real but short and flat Wikipedia tables, with row keys that are numeric indices in forward or reversed order (Section 1.4). Whether the tables appear in pretraining is unknown. No nonce-table control was run, so pretraining overlap and memorization cannot be excluded.

One causal regime. The main circuit-adequacy interventions use Instruct, markdown, row-index, chat zero-shot, where the random-head specificity controls stay inert. A diagnostic ablation also uses Llama base two-shot, where baseline row tracking is already marginal. In that diagnostic, twelve random heads raise wrong-row errors as much as the row tier does (Section 6.1). The causal attributions therefore hold where they are measured; the behavioral and observational results, which span the full 48-condition grid, are the ones with breadth.

Observational geometry is not intervened on. Table width, queried-column position, and header class are observed properties of a fixed corpus. The contrasts built on them are associations open to column-semantic and source-template confounding, and Chapter 8 marks every item that rests on them as provisional.

9.3.2 Sample and Metric Provenance

Outcome-conditioned pair samples. Aligned pairs are retained only when the model prefers the clean answer in the clean world and the alternative in the corrupt world, and only when a same-length substitution exists. Roughly half of attempted questions survive, and the losses are systematic, not random: the content axis loses candidates to alignment, the row axes to digit-per-token segmentation (Section 3.4). Every first-token result therefore describes the subpopulation of pairs meeting the alignment, distinct-first-token, and margin-reversal criteria, not the full task distribution. Patching uses a separately constructed sample with a 1300-token cap; attribution and faithfulness share a saved bank capped at 1024 tokens.

A first-token margin, not generated accuracy. Patching, attribution, and faithfulness all score a first-token logit margin; mean ablation scores full generation. The two disagree in a specific and reported way: the writer tier has a substantial tier-drop effect on the margin and no detectable full-generation cost (Section 6.5). This is a reason to read each number against its own metric rather than to merge them. That margin is also built on an answer token chosen by an outcome-conditioned rule: between the bare and space-prefixed tokenizations of the answer, the implementation keeps whichever the model assigns the higher logit (Section 3.4). A post hoc audit recovered the deterministic token (read from the tokenization of the prompt followed by the answer) for 4,738 of the 4,800 answer tokens in the saved banks (the remainder do not round-trip through decode and re-encode) and found four disagreements, all in Qwen’s corrupt answers; Appendix C tabulates the counts. Because the rule conditions on the measured logits, the margins may be marginally optimistic; nothing was re-run under the deterministic rule.

Balanced corruption axes and corpus prefixes. Faithfulness is an equal-weight macro-average over exactly 150 pairs from each corruption axis, not an estimate under a natural corruption distribution. Mean ablation uses the first 200 sorted corpus files, QK/OV decomposition the first 125, and patching scans sorted files until its quotas fill; these are deterministic prefixes rather than randomized table samples (Section 3.5).

9.3.3 Uncertainty and Controls

The random-head null is too sparse to calibrate. The ablation comparisons of Sections 6.1 and 6.3 are read against a single random head-set draw, and the minimality drops of Section 6.5.3 against a range of two. The controls are equal or larger, not exactly size-matched: twelve random heads stand against the four-head column tier, and sixteen in Llama or fifteen in Qwen against the twelve named heads, which makes them conservative for a necessity claim but means the comparison is not held at constant set size. A single draw shows only that the measured effect differs from what that one perturbation does; it does not say how much a comparable set *typically* costs, and it is not layer-matched, so it does not separate the contribution of a tier from the contribution of its depth. This bears hardest on Qwen, whose per-set ablation effects are small enough (≤ 0.015) that a calibrated null could plausibly cover them. The random controls in Chapter 6 are consequently sanity checks.

9.4 Future Work

The larger opportunity suggested by these results is not simply to test more variants of table lookup. It is to develop a principled way to design representations for model-facing structured data: identify landmarks that make latent structure addressable, verify the causal route by which a model uses them, and evaluate the representation jointly for accuracy, robustness, and inference cost. Row indices provide one controlled example of such a landmark. The experiments below are milestones toward a broader goal: a reusable account of structural addressing that can guide how data are presented to different models and tasks.

Make the mechanism predictive. Circuit discovery should be separated from confirmation. Head selection, thresholds, and analysis choices could be fixed on one bank and evaluated unchanged on held-out tables, questions, and corruption pairs. Nonce or content-permuted tables would help separate in-context structural processing from pretraining overlap, while distributions of many size- and depth-matched random head sets would replace the current sparse controls with calibrated comparisons. Together, held-out evaluation and a calibrated null would turn the present map into a falsifiable, out-of-sample claim that predicts which components matter rather than merely describing the data used to find them.

Build a theory of useful landmarks. Names, dates, non-contiguous identifiers, permuted integers, and nonce keys should be crossed with column position, row order, and exact versus paraphrased references in the question. Repeating the behavioral grid, routing measurements, and label-swap intervention would test whether content-addressable binding is specific to small integers or is a more general operation. The objective is not another ranking of formats, but a characterization of when a token remains a reliable structural handle as the surrounding surface form changes.

Use cue conflict as a diagnostic. The reversed-order two-shot penalty provides a controlled stress test because an additional prompt cue makes the indexed task worse. Sweeping the number and content of demonstrations across independently sampled banks, and filling the missing Qwen base zero-shot cell, would establish the effect’s scope. Measuring row-key QK margins and applying row-index swaps and retrieval patching under the same prompts would then distinguish a failure to bind the key from a later failure to use it. More broadly, this experiment asks how a model arbitrates between an explicit landmark in the data and a competing heuristic induced by demonstrations.

Test whether addressing composes. Atomic lookup isolates addressing, but aggregation, comparison, filtering, and multi-hop questions all require it before performing another operation. Tasks with known intermediate addresses could test whether matching and retrieval form a reusable front end for later arithmetic or comparison circuits, or whether each task constructs a new route. Hierarchical tables, repeated field names, nested JSON, and source-code scopes would extend the same question to structured sequences in which several landmarks have similar surface forms. Applying a fixed selection rule and held-out causal protocol across more model families and within-family size series would identify invariants at the level of operations rather than head coordinates, and would test whether the Llama–Qwen concentration difference is due to scale, architecture, or training.

Treat hints as computational interventions. Once useful landmarks can be identified, their value should be measured in computation as well as accuracy. A paired evaluation on reasoning models could present the same table–question instances with and without row indices and jointly measure input-token overhead, reported reasoning-token use, total generated tokens, completion latency, and estimated serving cost. Comparisons should also be conditioned on correct answers

so that short traces caused by premature failure are not counted as efficiency. The systems question is whether a small amount of explicit prompt structure can replace a larger generated search. At production volume, even modest reductions in sequential decoding could affect throughput and commercial serving cost.

Toward mechanism-aware structured interfaces. Row keys are only one possible scaffold. Schema summaries, explicit field identifiers, projected views, and stable names for nested objects could be evaluated under the same accuracy–robustness–cost tradeoff. A longer-term interface could choose a representation for the model, query, and workload: expose the landmarks needed for addressing, preserve the lexical anchors the model can bind, and omit irrelevant structure when doing so is safe. A mechanistic audit would test whether the model uses that representation as intended, and a runtime system could add a hint when its accuracy–cost tradeoff is favorable. On this view, row indices are a proof of concept rather than the endpoint: representations can act as computational scaffolds, and interpretability can help turn them into more reliable and efficient structured-reasoning systems.

Appendix A

Supplementary Behavioral Results

This appendix collects behavioral material that supports Chapter 4 but is not needed to follow its argument: the reversed-index figure on the original example bank, the complete demonstration-bank grid behind Table 4.1, and the marginal column-position analyses that motivated the exact-width reanalysis.

A.1 Reversed Row Order on the Original Example Bank

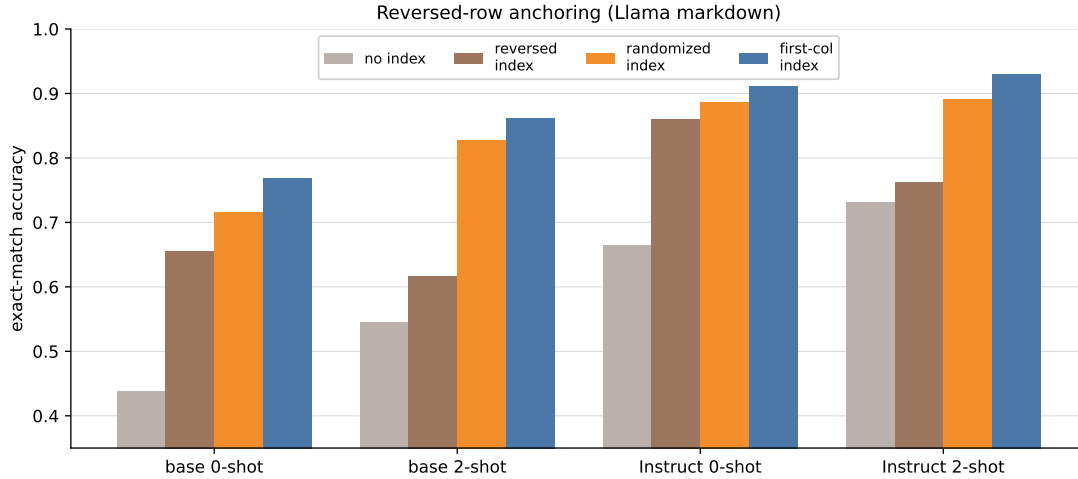


Figure A.1: Accuracy under the reversed-row condition (revidx) and the none, randidx, and rowid controls for Llama on markdown ($n \approx 20,100$ per cell). Two-shot results use the original three-row bank; zero-shot results are bank-independent. Reversal preserves the rowid > randidx > revidx ordering, but examples reduce accuracy only under revidx.

model	rowid			revidx		
	0-shot	2-shot		0-shot	2-shot	
		3-row	10-row		3-row	10-row
Llama base	0.768	0.862	0.825	0.655	0.616	0.617
Llama Instruct	0.912	0.930	0.915	0.860	0.763	0.799
Qwen base	0.871	0.874	0.861	—	0.798	0.775
Qwen Instruct	0.931	0.938	0.932	0.912	0.856	0.865

Table A.1: Exact-match accuracy for the ordinary (rowid) and reversed (revidx) row index at zero shots and at two shots under each demonstration bank, markdown, $n \approx 20,100$ per cell. The three-row bank is the main grid’s; in the ten-row bank both demonstration targets change rendered position under reversal. The two-shot inversion under reversal appears on both banks: switching banks recovers part of the deficit in Llama Instruct ($0.763 \rightarrow 0.799$), next to none in Llama base, and moves backwards in Qwen base, while shifting the ordinary rowid condition by -0.006 to -0.037 . The Qwen base reversed zero-shot cell was not run.

A.2 Marginal Column-Position Analyses

Figure A.2 reproduces the two column-position analyses that preceded the width-conditioned reanalysis of Section 4.4. They are kept here because they show what the confound looks like before it is removed, not because they support a claim.

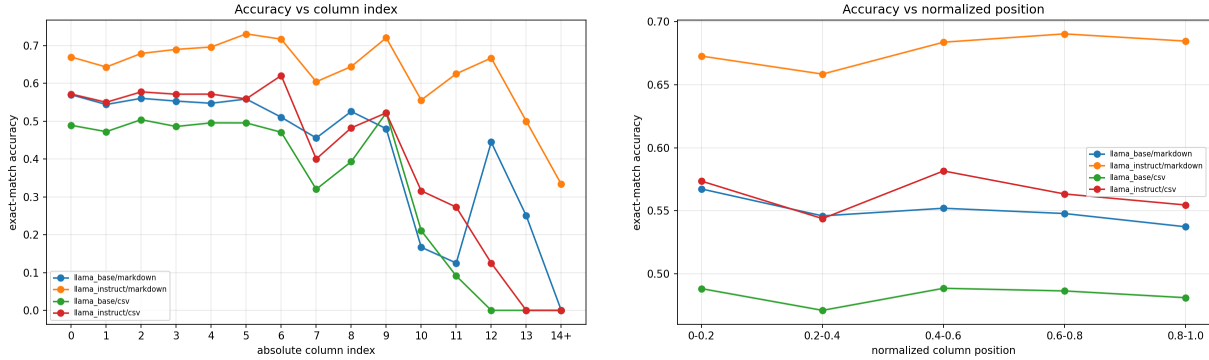


Figure A.2: Exact-match accuracy by absolute (left) and normalized (right) target-column position in four no-index Llama conditions: base two-shot and Instruct zero-shot on markdown and CSV ($n = 6,700$ each). Both panels pool over table width; the analysis in Section 4.4 conditions on width.

Appendix B

Observational Maps and Head Rankings

This appendix is the audit trail for candidate-head selection: the full per-head maps the selection rule of Section 3.4.5 read, and the top-ten rankings each roster tier was drawn from, so that the rosters of Table 3.3 can be checked against their sources.

Conventions, common to every panel: heatmaps index layers on the vertical axis with layer 0 at the bottom and heads on the horizontal axis; color scales are per-panel, so magnitudes should be read from the colorbars rather than compared across panels by hue, and where a panel carries a diverging scale extending below zero, the negative half is unused—every attention metric plotted here is nonnegative. Values are mean attention mass from the stated query position to the stated destination unless the panel is labeled structural precision (correct-cell mass as a fraction of the mass on the correct cell plus its row and column) or alignment (attention alignment against the causal-uniform baseline of Section 5.2).

B.1 Retrieval-Pool Maps

The retrieval tier is selected on final-token correct-cell attention alone (Figure 5.4). Figure B.1 shows what the same heads do with the rest of their mass, which is the evidence that the tier is reading a cell rather than a region.

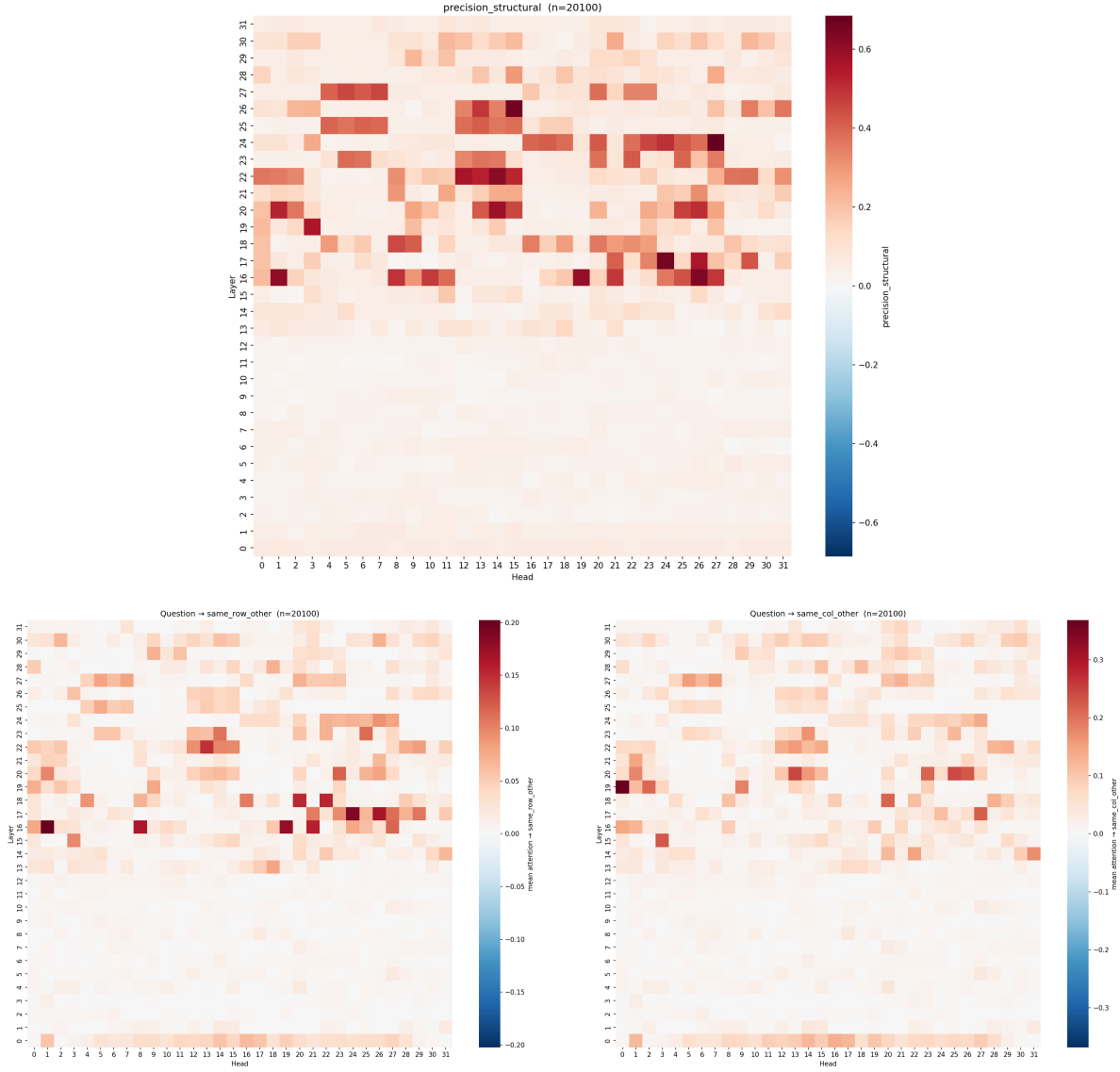


Figure B.1: Per-head structural precision (top) and final-token attention to other cells in the target row (bottom left) and column (bottom right), for Llama Instruct (markdown, zero-shot, rowid; $n = 20,100$).

B.2 Query-Token Routing Maps

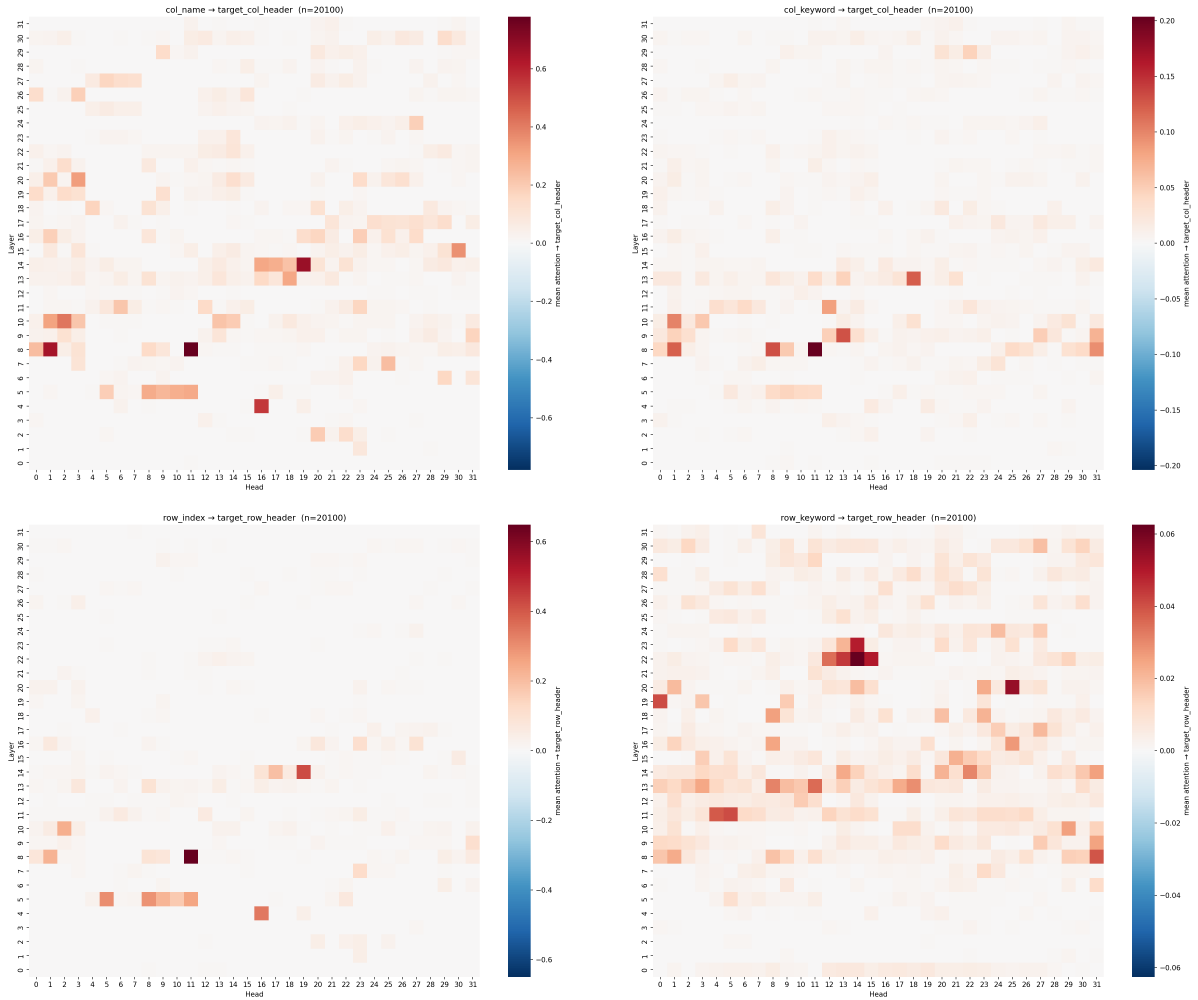


Figure B.2: The full per-head routing maps for Llama Instruct (markdown, zero-shot, rowid; $n = 20,100$). The left panels trace the question’s column name and row index to their table landmarks; the right panels use the generic words “column” and “row” as controls. Content-bearing tokens show sharper, more localized routing.

B.3 Cross-Family Replication Maps

Figure B.3 gives the three Qwen maps the selection rule read, so that the Qwen roster of Table 3.3 can be checked against its sources the same way the Llama roster can.

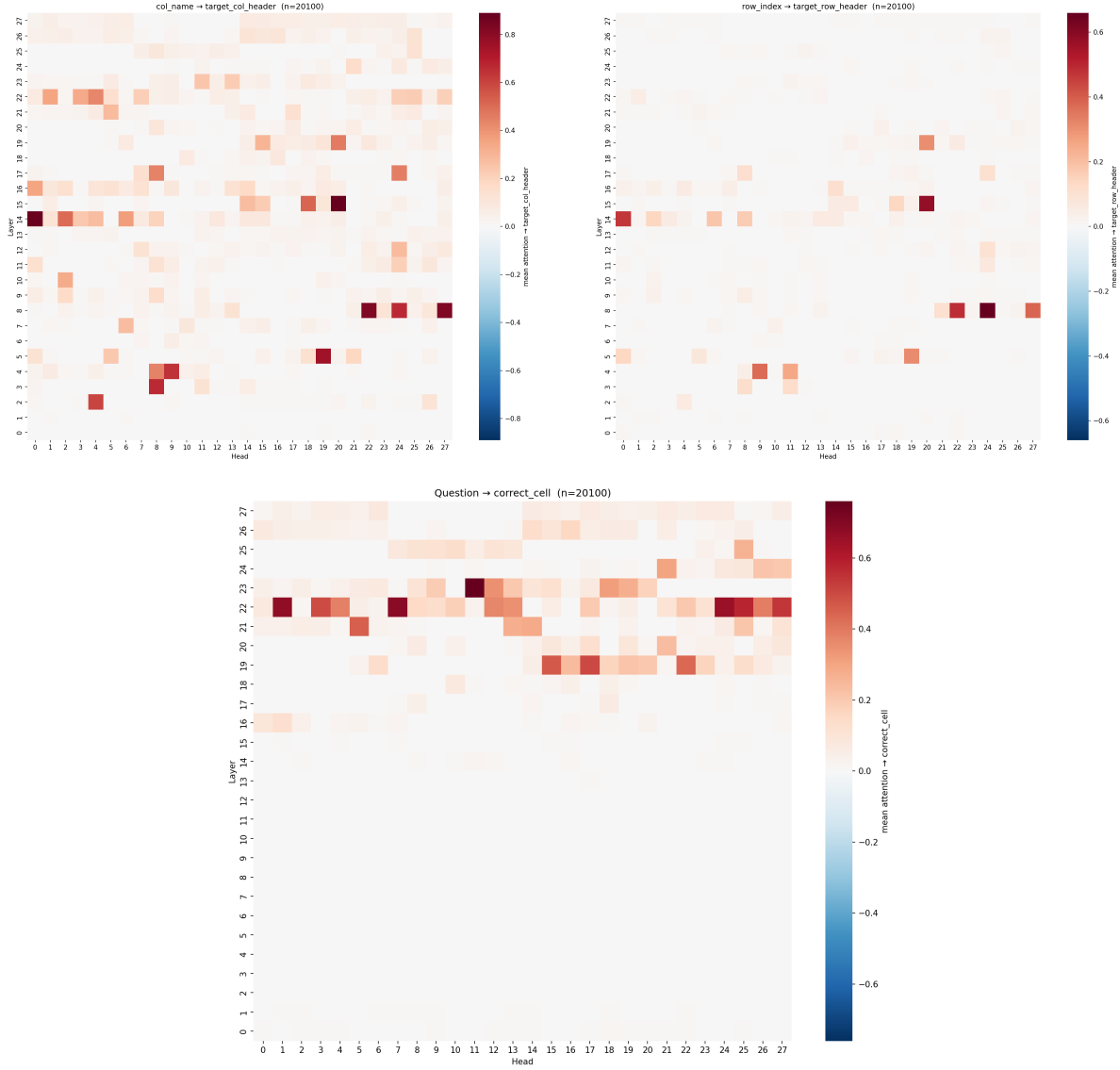


Figure B.3: The three roster-defining maps for Qwen Instruct (markdown, zero-shot, rowid; $n = 20,100$): column-name routing (top left), row-index routing (top right), and final-token correct-cell attention (bottom). The Llama signals recur at family-specific head coordinates.

B.4 Tuning Stability and Co-Localization

Figure B.4 is the per-metric form of the tuning-stability result quoted in Section 5.2. Figures B.5 and B.6 show how the metrics themselves co-localize across heads in all four models. Section 5.1 establishes their separation by depth.

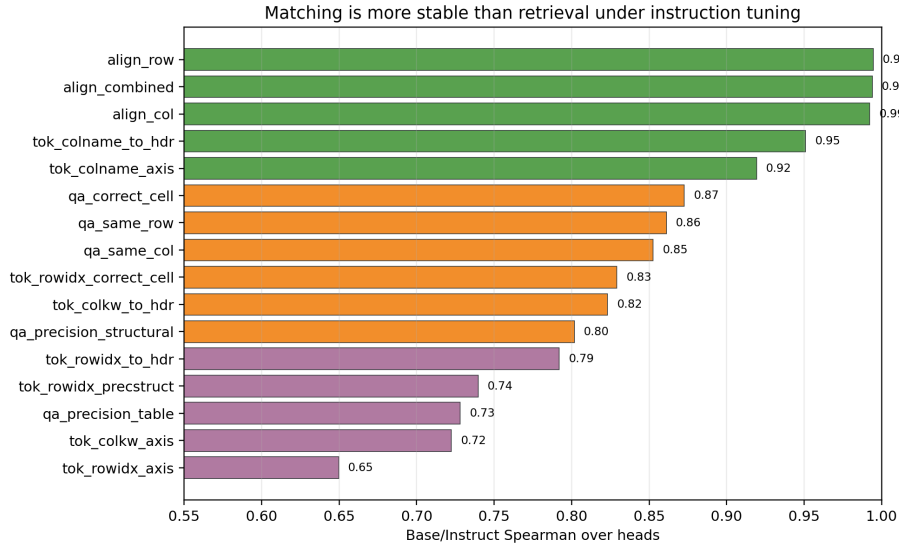


Figure B.4: Spearman rank correlation between Llama base and Instruct per-head maps. Alignment and column-name routing are more stable ($\rho = 0.95\text{--}0.99$) than correct-cell, structural-precision, and row-index-routing maps (0.79–0.87).

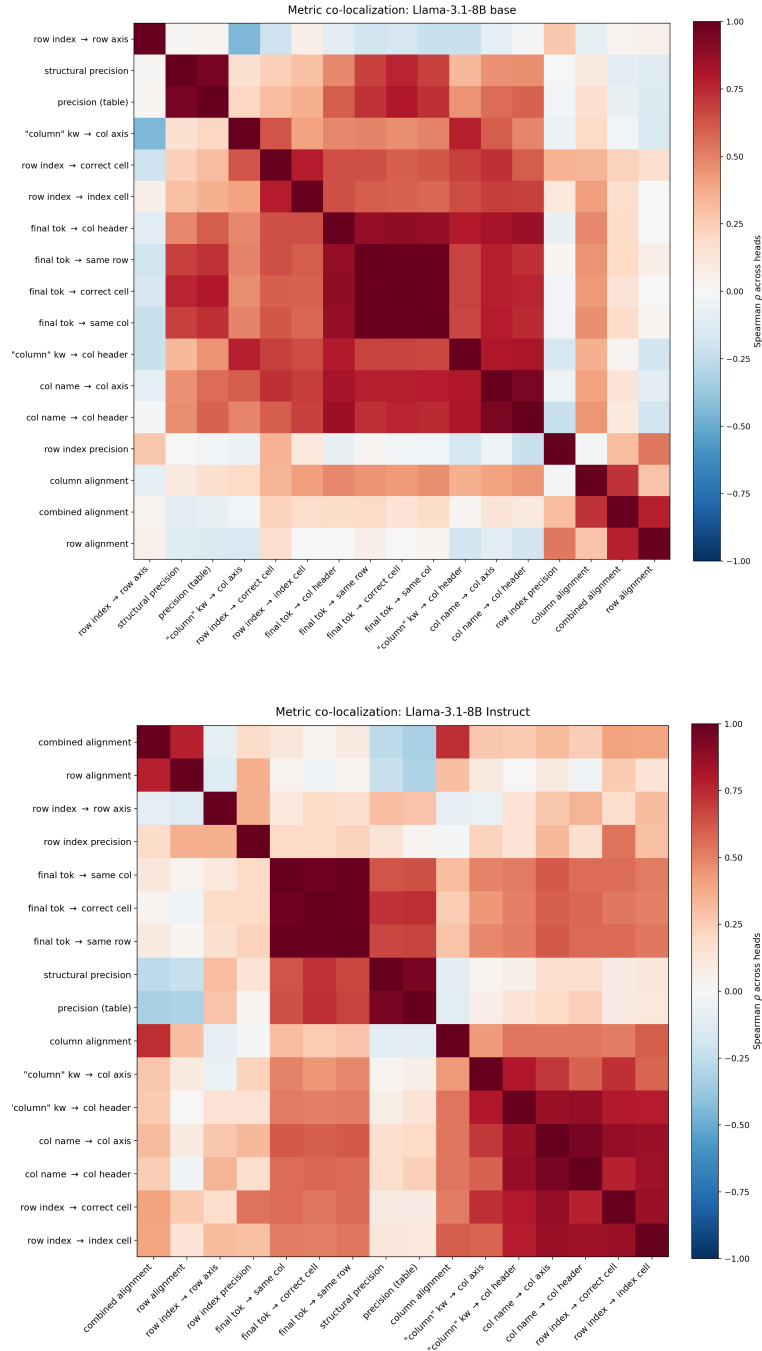


Figure B.5: Metric co-localization maps for Llama base (top) and Instruct (bottom), showing pairwise Spearman correlations among per-head metrics across 1,024 heads. Retrieval and question-routing measures form correlated sub-blocks within a broader structural-attention block; their positive cross-block correlations indicate overlapping rather than disjoint head sets.

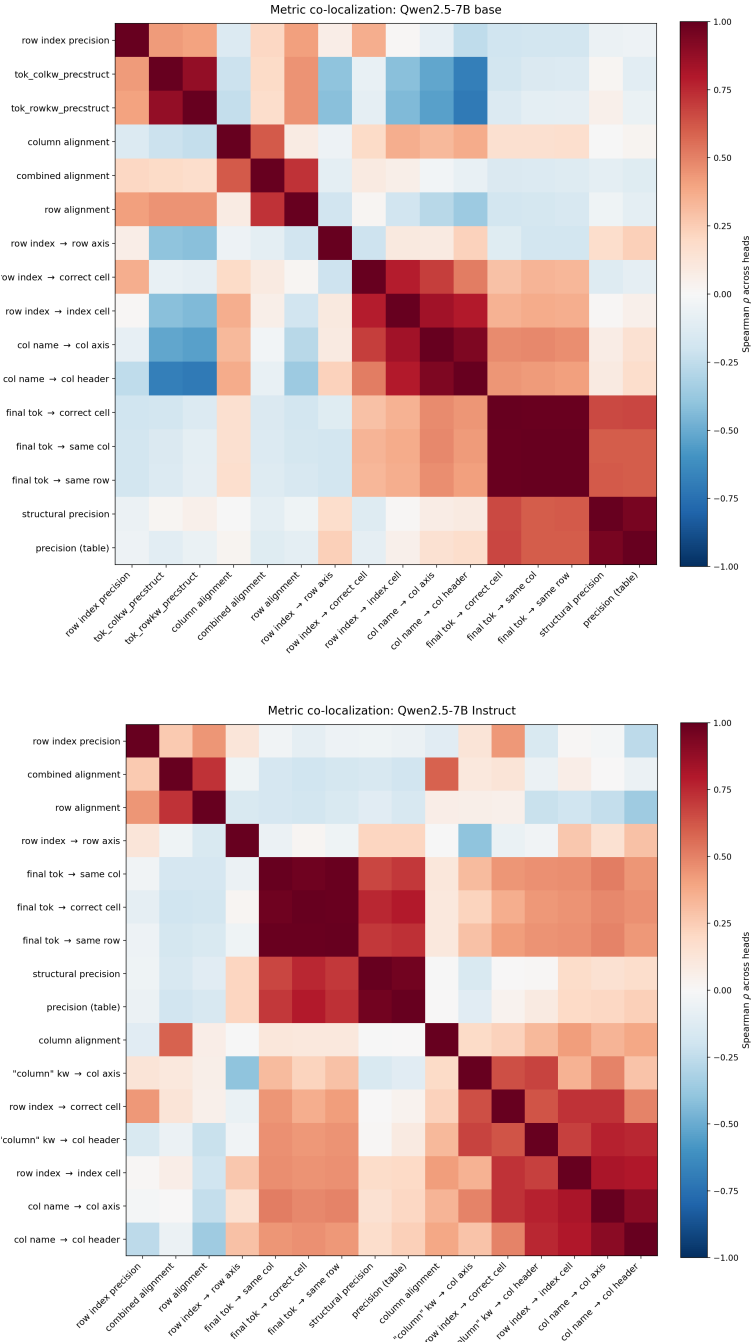


Figure B.6: The same maps for Qwen base (top) and Qwen Instruct (bottom), across 784 heads. The structural block and its retrieval and routing sub-blocks also appear in Qwen.

B.5 Top-Ten Rankings for the Primary Metrics

Tables B.1–B.3 list the ten highest-scoring heads on each primary selection metric for both Instruct models, with raw scores, since ranks alone hide whether adjacent heads are nearly tied or widely separated. A bullet marks membership in the tested rosters of Table 3.3. Only the metrics the selection rule reads are ranked. The generic-keyword and BOS destinations are omitted as *controls*: they exist to show that routing is content-selective, not to nominate heads, so ranking them would serve no selection purpose at any magnitude—the generic-column control in fact peaks at 0.20, well above the generic-row control’s 0.06 (Figure B.2).

Llama Instruct			Qwen Instruct	
rank	head	score	head	score
<i>column alignment</i>				
1	L8H11•	0.611	L8H22•	0.871
2	L4H16•	0.537	L8H24•	0.611
3	L21H2	0.509	L16H9	0.531
4	L6H3	0.476	L15H20•	0.510
5	L7H25	0.466	L4H9•	0.508
6	L5H22	0.440	L25H15	0.498
7	L27H28	0.374	L8H27•	0.482
8	L11H6	0.342	L4H11•	0.453
9	L14H19•	0.334	L8H25	0.451
10	L10H2•	0.328	L2H22	0.433
<i>row alignment</i>				
1	L4H25	0.269	L3H25	0.535
2	L4H30	0.253	L4H20	0.521
3	L0H13	0.208	L10H7	0.448
4	L0H0	0.190	L8H20	0.424
5	L0H9	0.182	L1H25	0.401
6	L2H11	0.171	L11H17	0.388
7	L5H2	0.168	L2H3	0.381
8	L3H10	0.151	L0H9	0.376
9	L2H0	0.144	L5H25	0.343
10	L4H27	0.130	L3H22	0.334

Table B.1: Top ten heads by column and row attention alignment (baseline-subtracted; Section 5.2). • member of the tested roster. The Llama maps use the row-index alignment corpus; Qwen’s alignment analysis was run only on the un-indexed markdown corpus, so its two columns are not condition-matched to Llama’s. No tested head appears in the row-alignment lists: row alignment is diffuse, and the row-matching tier is selected by row-index *routing* (Table B.2), not by row-tracking geometry (Section 5.2).

Llama Instruct			Qwen Instruct	
rank	head	score	head	score
column name → target header				
1	L8H11•	0.779	L15H20•	0.890
2	L14H19•	0.673	L14H0•	0.878
3	L8H1•	0.647	L8H27•	0.834
4	L4H16•	0.552	L8H22•	0.829
5	L10H2•	0.414	L5H19•	0.757
6	L15H30	0.355	L3H8	0.664
7	L20H3	0.325	L8H24•	0.661
8	L10H1	0.314	L4H9•	0.643
9	L14H16	0.303	L2H4	0.601
10	L13H18	0.301	L15H18	0.518
row index → target index cell				
1	L8H11•	0.649	L8H24•	0.659
2	L14H19•	0.421	L15H20•	0.579
3	L4H16•	0.341	L8H22•	0.489
4	L5H5	0.303	L14H0•	0.473
5	L5H8•	0.291	L8H27•	0.385
6	L5H11•	0.258	L4H9•	0.372
7	L10H2•	0.234	L5H19•	0.313
8	L8H1•	0.226	L19H20•	0.308
9	L5H9•	0.222	L4H11•	0.256
10	L14H17•	0.193	L14H6	0.183

Table B.2: Top ten heads by mean attention mass from the question’s content tokens to their table landmarks, markdown, zero-shot, rowid, $n = 20,100$. • member of the tested roster. The row-routing list makes the one selection-rule departure visible: rule-selected L5H5 (rank 4) was not tested, and tested L14H17 (rank 10) enters from the base-model maps (Section 3.4.5).

rank	Llama Instruct		Qwen Instruct	
	head	score	head	score
<i>final token $\rightarrow$ correct cell</i>				
1	L17H24•	0.582	L23H11•	0.760
2	L16H1•	0.533	L22H7•	0.688
3	L24H27•	0.501	L22H1•	0.677
4	L16H19	0.440	L22H24	0.658
5	L22H14	0.375	L22H25	0.582
6	L22H13	0.369	L22H27	0.542
7	L20H1	0.346	L19H17	0.502
8	L20H26	0.345	L22H3	0.495
9	L20H14	0.340	L19H15	0.468
10	L17H26	0.321	L21H5	0.452
<i>structural precision</i>				
1	L24H27•	0.685	L23H11•	0.805
2	L26H15	0.659	L22H7•	0.750
3	L17H24•	0.653	L22H25	0.735
4	L16H26	0.635	L22H24	0.726
5	L22H14	0.621	L22H1•	0.712
6	L16H1•	0.609	L22H27	0.668
7	L16H19	0.595	L22H26	0.644
8	L20H14	0.588	L22H13	0.632
9	L22H12	0.567	L21H5	0.622
10	L17H26	0.561	L23H12	0.618

Table B.3: Top ten heads by final-token correct-cell attention and by structural precision, mark-down, zero-shot, rowid, $n = 20,100$. • member of the tested roster. The retrieval tier is selected by correct-cell mass alone; structural precision is reported as corroboration. Both families’ tested trios are the top three on the selection metric, and mass is spread over a band of further heads rather than concentrated, the observational diffuseness discussed in Section 5.4.

Appendix C

Circuit Robustness and Measurement Details

This appendix tabulates the numbers behind the faithfulness figures of Section 6.5.1 and the answer-token audit summarized in Section 9.3.2.

C.1 Faithfulness Grids

Tables C.1 and C.2 give the values plotted in Figure 6.7, one row per grid point, so that the crossings quoted in Section 6.5.1 can be checked against the two grid points they interpolate between. Table C.3 resolves the same sweep by corruption axis, the weighting-sensitivity check that the pooled macro-average conceals.

K	attribution f [95% CI]	named-first	random	inverse	mean
0	0.000 [0.000, 0.000]	0.000	0.000–0.000	1.000	0.477
1	0.006 [0.005, 0.007]	0.000	0.000–0.000	0.643	0.477
2	0.037 [0.031, 0.044]	0.000	0.000–0.000	0.412	0.478
5	0.030 [0.025, 0.036]	0.000	0.000–0.002	0.318	0.475
12	0.103 [0.087, 0.120]	0.063	0.000–0.002	0.355	0.475
16	0.099 [0.083, 0.114]	0.140	0.000–0.002	0.493	0.477
26	0.087 [0.073, 0.102]	0.123	0.000–0.001	0.386	0.476
51	0.121 [0.106, 0.135]	0.178	0.001–0.073	0.209	0.479
102	0.390 [0.365, 0.415]	0.427	0.002–0.075	0.098	0.479
205	0.740 [0.719, 0.760]	0.745	0.006–0.103	0.016	0.484
512	0.991 [0.987, 0.996]	0.991	0.190–0.556	−0.001	0.492
1024	1.000 [1.000, 1.000]	1.000	1.000–1.000	0.000	1.000

Table C.1: Llama Instruct interchange faithfulness f at each circuit size K (of 1,024 heads; 600 pairs, 150 per axis). Columns: the attribution ranking with its pair-bootstrap 95% CI; the observationally fixed named-first ranking; the range over three random-ranking seeds; the inverse (complement) circuit, whose collapse is the inverse-complement necessity diagnostic of Section 6.5.2; and the mean-ablation flavor, whose near-flat profile is the flavor-sensitivity result of Section 6.5.4. The circuit-performance ratio (CPR) summarizes a whole column as the trapezoidal area under its curve plotted against the head fraction K/H [26], so a ranking that reaches high faithfulness at small K scores near 1 while one that recovers only near full size scores near 0: attribution 0.831, named-first 0.838, random 0.328–0.501, inverse 0.033, mean flavor 0.615.

K	attribution f [95% CI]	named-first	random	inverse	mean
0	0.000 [0.000, 0.000]	0.000	0.000–0.000	1.000	0.486
1	0.000 [0.000, 0.000]	0.000	0.000–0.000	0.357	0.487
2	0.000 [0.000, 0.000]	0.000	0.000–0.000	0.377	0.486
5	0.057 [0.048, 0.068]	0.000	0.000–0.000	0.279	0.487
12	0.076 [0.063, 0.089]	0.000	0.000–0.000	0.273	0.485
16	0.098 [0.082, 0.115]	0.065	0.000–0.000	0.222	0.486
26	0.115 [0.097, 0.134]	0.103	0.000–0.000	0.202	0.488
51	0.199 [0.176, 0.225]	0.192	0.000–0.000	0.020	0.487
102	0.463 [0.436, 0.493]	0.475	0.000–0.011	0.003	0.485
205	0.901 [0.891, 0.911]	0.903	0.001–0.029	0.000	0.486
512	0.995 [0.992, 0.999]	0.996	0.026–0.395	0.000	0.484
784	1.000 [1.000, 1.000]	1.000	1.000–1.000	0.000	1.000

Table C.2: Qwen Instruct interchange faithfulness f at each circuit size K (of 784 heads; 600 pairs, 150 per axis), columns as in Table C.1. Values of −0.000 are rounded to 0.000. CPR: attribution 0.836, named-first 0.836, random 0.183–0.328, inverse 0.013, mean flavor 0.574.

K	Llama Instruct				Qwen Instruct			
	a	b	c	d	a	b	c	d
0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
1	0.000	0.000	0.025	0.000	0.000	0.000	0.000	0.000
2	0.000	0.000	0.150	0.000	0.000	0.000	0.000	0.000
5	0.000	0.000	0.121	0.000	0.000	0.000	0.230	−0.001
12	0.001	0.000	0.414	−0.001	0.000	0.000	0.304	−0.002
16	0.001	0.000	0.395	−0.002	0.000	0.000	0.395	−0.002
26	0.000	0.000	0.351	−0.002	0.000	0.000	0.463	−0.002
51	0.011	0.042	0.376	0.055	0.006	0.078	0.605	0.108
102	0.068	0.434	0.663	0.394	0.128	0.440	0.828	0.459
205	0.461	0.803	0.903	0.793	0.870	0.903	0.926	0.904
512	0.982	0.995	0.996	0.992	0.985	0.998	0.999	1.001
all	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000

Table C.3: Interchange faithfulness under the attribution ranking resolved by corruption axis, 150 pairs per axis per model: (a) column name, (b) row number, (c) cell content, (d) row-index swap (Section 3.4). The final row is the full model ($K = 1024$ for Llama, 784 for Qwen). The cell-content axis recovers first in both families; the localization axes (a, b, d) stay near zero until $K \approx 51$ –102 and cross late, the pattern discussed in Section 6.5.1.

C.2 Answer-Token Audit

Table C.4 gives the counts behind the answer-token limitation of Section 9.3.2, separating how often the outcome-conditioned rule could have mattered from how often it actually disagreed with the deterministic tokenization.

	Llama Instruct		Qwen Instruct	
	clean	corrupt	clean	corrupt
answer tokens in the pair bank	1200	1200	1200	1200
bare/spaced tokenizations differ	1200	1200	1200	1200
rule keeps the bare token	1196	1200	1200	1196
deterministic token recovered	1167	1171	1200	1200
recovered token disagrees with rule	0	0	0	4

Table C.4: The audit behind the answer-token limitation of Section 9.3.2. For every clean and corrupt answer in the saved pair banks, the outcome-conditioned rule chose between the bare and space-prefixed tokenization; the audit re-derived the deterministic token from the tokenization of prompt followed by answer. It was recoverable for 4,738 of 4,800 answers (the remainder do not round-trip through decode and re-encode) and disagreed with the rule’s choice four times, all in Qwen’s corrupt answers.

Bibliography

- [1] Yihan Cao, Shuyi Chen, Ryan Liu, Zhiruo Wang, and Daniel Fried. API-assisted code generation for question answering on varied table structures. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 14536–14548, 2023. 2.2
- [2] Wenhui Chen. Large language models are few(1)-shot table reasoners. In *Findings of the Association for Computational Linguistics: EACL*, pages 1120–1130, 2023. 2.2
- [3] Wenhui Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyu Zhou, and William Yang Wang. TabFact: A large-scale dataset for table-based fact verification. In *International Conference on Learning Representations (ICLR)*, 2020. arXiv:1909.02164. 2.1
- [4] Wenhui Chen, Hanwen Zha, Zhiyu Chen, Wenhan Xiong, Hong Wang, and William Yang Wang. HybridQA: A dataset of multi-hop question answering over tabular and textual data. In *Findings of the Association for Computational Linguistics: EMNLP*, pages 1026–1036, 2020. arXiv:2004.07347. 2.1
- [5] Arthur Conmy, Augustine N. Mavor-Parker, Aengus Lynch, Stefan Heimersheim, and Adrià Garriga-Alonso. Towards automated circuit discovery for mechanistic interpretability. In *Advances in Neural Information Processing Systems*, 2023. arXiv:2304.14997. 2.3
- [6] Xiang Deng, Huan Sun, Alyssa Lees, You Wu, and Cong Yu. TURL: Table understanding through representation learning. *Proceedings of the VLDB Endowment*, 14(3):307–319, 2020. doi: 10.14778/3430915.3430921. 2.1
- [7] Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, et al. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 2021. <https://transformer-circuits.pub/2021/framework/index.html>. 2.3, 3.4.4
- [8] Atticus Geiger, Hanson Lu, Thomas Icard, and Christopher Potts. Causal abstractions of neural networks. In *Advances in Neural Information Processing Systems*, 2021. arXiv:2106.02997. 2.3
- [9] Michael Hanna, Sandro Pezzelle, and Yonatan Belinkov. Have faith in faithfulness: Going beyond circuit overlap when finding model mechanisms. In *Conference on Language Modeling (COLM)*, 2024. arXiv:2403.17806. 2.3, 3.4.4
- [10] Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Müller, Francesco Piccinno, and Julian Martin Eisenschlos. TaPas: Weakly supervised table parsing via pre-training. In

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL), pages 4320–4333, 2020. arXiv:2004.02349. 2.1

- [11] Sirui Huang, Yanggan Gu, Zhonghao Li, Xuming Hu, Li Qing, and Guandong Xu. Struct-Fact: Reasoning factual knowledge from structured data with large language models. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 7521–7552, Vienna, Austria, July 2025. Association for Computational Linguistics. 2.2
- [12] Binyuan Hui, Jian Yang, Zeyu Cui, et al. Qwen2.5-Coder technical report. *arXiv preprint arXiv:2409.12186*, 2024. 7.4
- [13] Hiroshi Iida, Dung Thai, Varun Manjunatha, and Mohit Iyyer. TABBIE: Pretrained representations of tabular data. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 3446–3456, 2021. 2.1
- [14] Mohit Iyyer, Wen-tau Yih, and Ming-Wei Chang. Search-based neural structured learning for sequential question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1821–1831, 2017. 2.1
- [15] Jinhao Jiang, Kun Zhou, Zican Dong, Keming Ye, Wayne Xin Zhao, and Ji-Rong Wen. StructGPT: A general framework for large language model to reason over structured data. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9237–9251, 2023. 2.2
- [16] Zhengbao Jiang, Yi Mao, Pengcheng He, Graham Neubig, and Weizhu Chen. OmniTab: Pretraining with natural and synthetic data for few-shot table-based question answering. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 932–942, 2022. 2.1
- [17] Najoung Kim, Sebastian Schuster, and Shubham Toshniwal. Code pretraining improves entity tracking abilities of language models. *arXiv preprint arXiv:2405.21068*, 2024. 7.4
- [18] János Kramár, Tom Lieberum, Rohin Shah, and Neel Nanda. AtP*: An efficient and scalable method for localizing LLM behaviour to components. *arXiv preprint arXiv:2403.00745*, 2024. 2.3
- [19] Kenneth Li, Aspen K. Hopkins, David Bau, Fernanda Viégas, Hanspeter Pfister, and Martin Wattenberg. Emergent world representations: Exploring a sequence model trained on a synthetic task. In *International Conference on Learning Representations (ICLR)*, 2023. arXiv:2210.13382. 1.1, 2.3, 9.2
- [20] Liyao Li, Jiaming Tian, Hao Chen, Wentao Ye, Chao Ye, Haobo Wang, Ningtao Wang, Xing Fu, Gang Chen, and Junbo Zhao. LongTableBench: Benchmarking long-context table reasoning across real-world formats and domains. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 11927–11965, Suzhou, China, November 2025. Association for Computational Linguistics. 2.2
- [21] Qian Liu, Bei Chen, Jiaqi Guo, Morteza Ziyadi, Zeqi Lin, Weizhu Chen, and Jian-Guang Lou. TAPEX: Table pre-training via learning a neural SQL executor. In *International Conference on Learning Representations (ICLR)*, 2022. 2.1

- [22] Tianyang Liu, Fei Wang, and Muhao Chen. Rethinking tabular data understanding with large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, pages 450–482, 2024. 1.1, 2.2, 8
- [23] Llama Team, AI @ Meta. The Llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. 1.2, 3.3, 7.4
- [24] Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. In *Advances in Neural Information Processing Systems*, 2022. arXiv:2202.05262. 2.3, 3.4.3
- [25] Joseph Miller, Bilal Chughtai, and William Saunders. Transformer circuit faithfulness metrics are not robust. In *Conference on Language Modeling (COLM)*, 2024. arXiv:2407.08734. 2.3, 3.4.3, 3.4.4, 6.5.4
- [26] Aaron Mueller, Atticus Geiger, Sarah Wiegrefe, Dana Arad, Iván Arcuschin, Adam Belfki, Yik Siu Chan, Jaden Fiotto-Kaufman, Tal Haklay, Michael Hanna, et al. MIB: A mechanistic interpretability benchmark. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, volume 267 of *Proceedings of Machine Learning Research*, pages 45069–45108. PMLR, 2025. arXiv:2504.13151. 2.3, 3.4.4, 3.4.4, C.1
- [27] Linyong Nan, Chiachun Hsieh, Ziming Mao, Xi Victoria Lin, Neha Verma, Rui Zhang, Wojciech Kryściński, Hailey Schoelkopf, Riley Kong, Xiangru Tang, et al. FeTaQA: Free-form table question answering. *Transactions of the Association for Computational Linguistics (TACL)*, 10:35–49, 2022. 2.1
- [28] Neel Nanda and Joseph Bloom. TransformerLens. <https://github.com/TransformerLensOrg/TransformerLens>, 2022. 1.2, 3.3, 3.5
- [29] Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, et al. In-context learning and induction heads. *Transformer Circuits Thread*, 2022. arXiv:2209.11895. 1.1, 2.3, 7.2, 9.2
- [30] Panupong Pasupat and Percy Liang. Compositional semantic parsing on semi-structured tables. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (ACL-IJCNLP)*, pages 1470–1480, 2015. arXiv:1508.00305. 2.1, 3.2
- [31] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32, pages 8024–8035, 2019. 3.5
- [32] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine*

Learning Research, 12:2825–2830, 2011. 4.4

- [33] Jackson Petty, Sjoerd van Steenkiste, and Tal Linzen. How does code pretraining affect language model task performance? *arXiv preprint arXiv:2409.04556*, 2024. 7.4
- [34] Qwen Team. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024. 1.2, 3.3, 7.4
- [35] Yuan Sui, Mengyu Zhou, Mingjie Zhou, Shi Han, and Dongmei Zhang. Table meets LLM: Can large language models understand structured table data? a benchmark and empirical study. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining (WSDM)*, pages 645–654, 2024. 1.1, 2.2, 8
- [36] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 3319–3328. PMLR, 2017. 2.3
- [37] Aaquib Syed, Can Rager, and Arthur Conmy. Attribution patching outperforms automated circuit discovery. In *Proceedings of the 7th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 407–416, 2024. arXiv:2310.10348. 2.3
- [38] Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart M. Shieber. Investigating gender bias in language models using causal mediation analysis. In *Advances in Neural Information Processing Systems*, 2020. arXiv:2004.12265. 2.3
- [39] Kevin Ro Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. Interpretability in the wild: a circuit for indirect object identification in GPT-2 small. In *International Conference on Learning Representations (ICLR)*, 2023. arXiv:2211.00593. 1.1, 2.3, 3.4.3, 3.4.3, 3.4.3, 3.4.4, 3.4.4, 6.3
- [40] Zhiruo Wang, Haoyu Dong, Ran Jia, Jia Li, Zhiyi Fu, Shi Han, and Dongmei Zhang. TUTA: Tree-based transformers for generally structured table pre-training. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*, pages 1780–1790, 2021. 2.1
- [41] Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. TaBERT: Pretraining for joint understanding of textual and tabular data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 8413–8426, 2020. 2.1
- [42] Fred Zhang and Neel Nanda. Towards best practices of activation patching in language models: Metrics and methods. In *International Conference on Learning Representations (ICLR)*, 2024. arXiv:2309.16042. 2.3, 3.4.3, 3.4.3
- [43] Xiaokang Zhang, Sijia Luo, Bohan Zhang, Zeyao Ma, Jing Zhang, Yang Li, Guanlin Li, Zijun Yao, Kangli Xu, Jinchang Zhou, et al. TableLLM: Enabling tabular data manipulation by LLMs in real office usage scenarios. In *Findings of the Association for Computational Linguistics: ACL*, pages 10315–10344, 2025. 2.2
- [44] Xuanliang Zhang, Dingzirui Wang, Keyan Xu, Qingfu Zhu, and Wanxiang Che. How do language models understand tables? a mechanistic analysis of cell location. *arXiv preprint*

arXiv:2602.08548, 2026. 2, 2.4, 6.2, 7, 7.1, 7.4

- [45] Victor Zhong, Caiming Xiong, and Richard Socher. Seq2SQL: Generating structured queries from natural language using reinforcement learning. *arXiv preprint arXiv:1709.00103*, 2017. 2.1