How Do Language Models Route Information Through Repeated Tokens?

Anonymous authors
Paper under double-blind review

Abstract

Repeated words are common in natural-language inputs to Language Models (LMs). We assume they serve as the anchors where the contextual information is collected and stored, and thereby detected and forwarded to the later repetition, by a set of attention heads named Duplicate Token Heads (DTHs). Therefore, we investigate such attention heads and find that: **(1)** Not all DTHs necessarily route information that is causally relevant to a given task since only a subset of attention heads can access the appropriate task subspace and route that information. **(2)** DTHs do not identify information-routing targets through simple input-token matching alone; contextual information also plays an important role. **(3)** DTHs become increasingly important as context length grows: ablating them causes substantially greater accuracy degradation in long contexts than in short ones, suggesting that direct local processing can serve as a competitive fallback only for short contexts, whereas long-context inference increasingly relies on anchor-based information. Our findings provide preliminary evidence that LMs utilize a cache-and-forward mechanism for information from earlier context, rather than re-traversing preceding tokens at later positions to retrieve that information.

1 Introduction

In natural language sentence, some words are often repeated. For example, names, objects, and identifiers often appear multiple times in one sentence and are referred to again later. Such words can be important anchors for "caching" contextual information during both human language processing and also computational language modeling (e.g., Large **L**anguage **M**odels (LMs)), which is shown in prior work (Wang et al., 2023; Prakash et al., 2024; 2026; Leviathan et al., 2025). However, the detailed mechanism in LMs for **locating** and **routing** such information remains unclear.

Previous works (Wang et al., 2023) find that the repeated structures are handled by **D**uplicate **T**oken **H**eads (DTHs), which exhibit a strong attention preference for earlier occurrences of the same token. Such heads can read information from an earlier repeated position and write it to a later occurrence (Wang et al., 2023; Prakash et al., 2024; Tang et al., 2026; Huang et al., 2024). However, prior studies mainly consider simple settings in which single-token identity is sufficient to determine the relevant source, e.g, in "The card is in Box C, the bell is in Box O, ..., Box C contains the", the token "C" uniquely identifies the earlier occurrence "Box C", with no other "C" acting as distractors. However, realistic inputs are often more ambiguous, e.g., "The card is in Box C, the bell is in Drawer C, ..., Box C contains the", repeated tokens may occur in multiple contexts, and token identity alone may not uniquely determine which earlier occurrence contains the relevant information. In this work, we study how DTHs operate under such more complex and realistic conditions.

To this end, we re-analyze the mechanism of DTHs. In detail, **(1, §2.3, 2.4) Identifying Functionally Significant DTHs.** To narrow the scope of our investigation (and also find what information is carried by the DTH), we first identify DTHs with significant causal effects by directly extracting the outputs of these DTHs as their information payloads and decode their contents using activation patching, and we find that: **DTHs Do not Necessarily Imply Effective Information.** Not all DTH outputs can effectively improve output accuracy when patched into a bare-query run whose required information is available only in a separate run where the patched outputs are extracted. Instead, strong improvement are concentrated in a subset of DTHs, and we can identify a subset of DTHs

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

(named functional-DTHs) with significant effects on the output and conduct subsequent experiments on them.

(2, §3) QK Matrix Analysis for the Addressing Mechanism: We argue that DTH addressing is not based solely on single-token matching (i.e., only read and match the information from the bottom of the residual stream (token embedding)), but contextual information from the residual branch in these tokens' hidden states, particularly information about their prefixes, also modulates which earlier occurrence is selected, to correctly solve the information source in scenarios like the aforementioned "Drawer C". As illustrated in Fig. 1, such processing is based on two-step operation: **first**, previous layers inject contextual information to the previous and current occurrence of the repeated token; **then**, the DTHs identify such contextual information apart from the token embedding propagated from the residual stream, by their QK matrices. So that they can locate both the similar token and similar context environment.

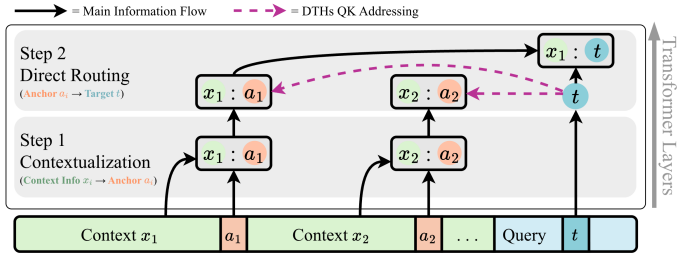

Figure 1: **Direct routing.** Step 1, contextualization: earlier computation encodes contextual information x_i into the representation at anchor a_i. Step 2, direct routing: the query–key (QK) interactions of functional-DTHs address earlier anchors from target t, read information from a matching source anchor, and write it to t. Anchors are earlier repeated token positions that can carry contextual information; the target is the repeated token position in the query that receives information. A source anchor is an anchor from which functional-DTHs route information to a particular target.

(3, §4) DTHs Work More Effective in Longer Context. In addition, we report an interesting finding regarding the function of DTHs in §4. Intuitively, the longer the context, the more the model should rely on such locatable anchors to carry forward the large amount of contextual information. We find that this is indeed the case by simply ablating the DTH from the computational graph, and observe that: Longer contexts suffer more severe accuracy degradation. However, shorter contexts do not suffer substantial ablation-induced degradation, suggesting a competitive fallback mechanism based on more direct local processing without information relay; this mechanism, however, breaks down in long contexts. This finding supports our earlier motivation: these repeated characters serve as anchors for preserving and compressing contextual information, and thus play an important role in language-model computation.

The main contributions of this paper are: (1) We show that only a subset of DTHs are causally involved in transmitting task-relevant information, separating attention similarity from functional information routing. **(2)** We demonstrate that these functional-DTHs resolve ambiguous repeated tokens through context-conditioned QK addressing, rather than simple token-identity matching. **(3)** We uncover a contextualize-then-route computation: earlier layers encode contextual information into repeated-token anchors, which are subsequently used by DTHs as addressable carriers for direct information transfer. This mechanism becomes increasingly important with longer contexts, suggesting that repeated tokens provide an efficient intermediate routing structure for long-range computation.

2 Preparation

2.1 Duplicate token heads

Duplicate token heads preferentially attend to earlier positions containing the same token. They were named in a study of the indirect object identification (IOI) circuit (Wang et al., 2023). For an input such as "`When Mary and John went to the store. John gave a drink to`", these heads were interpreted as passing positional signals about the repeated name to downstream components, helping the model exclude the repeated subject and identify the correct indirect object. Subsequent work found that components from the IOI circuit, including DTHs, are reused in the Colored Objects task (Merullo et al., 2024). Further analysis with InversionView found that

some DTH outputs encode the name's token identity rather than the previously hypothesized positional information (Huang et al., 2024). Together, these findings indicate that similar repeated-token addressing patterns need not imply the same transmitted content. The information-transfer functions and underlying mechanisms of DTHs therefore remain incompletely understood.

We study how direct routing transfers information through repeated tokens from three perspectives: its components, addressing mechanism, and causal role. We first identify DTHs that transfer information from earlier repeated token positions, then examine how these heads locate the relevant anchors, and finally test how blocking this path affects task performance.

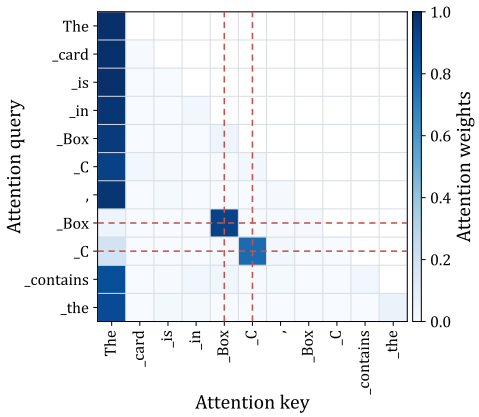

Figure 2: Attention map of a common DTHs (Qwen3-8B, L19H26). The intersections of the red dashed lines mean the DTH preferentially attends to earlier positions containing the same token.

2.2 Experimental Settings

Models. We study six language models from three families: Llama-3.2-1B and Llama-3.1-8B (Grattafiori et al., 2024; Meta, 2024); Qwen3-8B, Qwen3-14B, and Qwen3-32B (Qwen Team, 2025); and OLMo-2-7B (Team OLMo et al., 2024). All experiments use publicly available model weights without additional task-specific fine-tuning. We use plain-text completion prompts rather than chat templates throughout. Unless otherwise specified, we present detailed analyses using Qwen3-8B.

Tasks. Our main experiments cover three information-binding task families: **entity tracking** (Kim & Schuster, 2023; Dai et al., 2024; Prakash et al., 2024), **variable binding** (Wu et al., 2025), and **spatial reasoning** (Weston et al., 2015; Kuratov et al., 2024). We also examine two types of ICL tasks: **classification**, comprising SST-2 and SST-5 (Socher et al., 2013), AGNews (Zhang et al., 2015), and FP (Malo et al., 2013); and **generation**, comprising English–French translation (Todd et al., 2024; Conneau et al., 2017), Present-past and Present-ing transformations, and antonym and synonym prediction (Hendel et al., 2023; Todd et al., 2024). Unless otherwise specified, we present detailed analyses in entity tracking task. Appendix A.1 describes the datasets and input construction.

2.3 Identifying Duplicate Token Heads

We first discover DTH candidates from their repeated-token attention patterns, then (in §2.4) identify functional-DTHs through single-head value patching, and test their group information-transfer effects on independent data. We then compare the functional sets across tasks to assess component reuse.

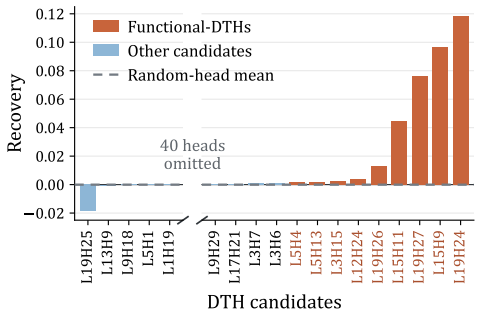

Figure 3: The result of single-head value patching recovery. Heads are sorted by recovery, with 40 near-zero-recovery heads in the middle omitted for clarity. The selected functional-DTHs are highlighted in orange and remain visible.

Discovering DTH candidates from repeated token preference. To avoid making candidate discovery depend on a particular task format, we construct a shared repeated-token discovery dataset, from English Wikipedia articles in GoodWiki (Choi, 2023) and questions and solutions in GSM8K (Cobbe et al., 2021). These texts contain naturally occurring repetitions of words, symbols, and numbers, enabling us to observe repeated-token addressing beyond a single task. We quantify this preference using the **DTH score**: the ratio of the mean attention weight assigned to earlier positions with the same token to the mean assigned to other eligible earlier positions, excluding attention-sink positions (Gu et al., 2025). We rank all attention heads in each model by this score and retain the top 5% as candidates (see Appendix D.1 for the DTH-score rankings). This candidate pool remains fixed across downstream tasks; its information-transfer function is established through the following interventions.

2.4 IDENTIFYING FUNCTIONAL DUPLICATE TOKEN HEADS

Identifying functional components through single-head value patching. Attending to an anchor does not establish that a head transfers information from it that affects the answer. Therefore, not all DTHs have a significant effect on the output, and for efficiency, we would like to restrict our subsequent experiments to those DTHs that do (named functional DTH). We therefore conduct activation patching, i.e., copy the DTH outputs from a contexted run (which potentially include context information necessary for the task) to a un-contexted run, and evaluate the output on the patched un-contexted run (patching details see Appendix A.2). Such experiment is done for each candidate on each task and rank candidates by recovery toward the donor answer. As a control, we apply the same single-head intervention to random heads sampled outside the candidate pool, matching the number and layer distribution of all candidates. Figure 3 shows the entity-tracking results for Qwen3-8B. Strong positive recovery is concentrated in a subset of DTH candidates. Other candidates have weaker effects, some negative, while the mean effect of random heads is close to zero. Thus, an attentional preference for repeated tokens alone is insufficient to establish an information-transmission function; only a small subset of DTHs transmit causally relevant information. Therefore, we retain the top $\lceil 0.15K \rceil$ heads as that model's **functional-DTHs** for the following experiments.

Table 1: Results of group value patching across models and tasks. Recovery quantifies the fraction of the donor–recipient answer-preference gap recovered by value patching. ET: entity tracking; VB: variable binding; SR: spatial reasoning.

Model	Task	Functional-DTHs		Random heads
		Anchors	Non-anchors	Anchors
Llama 3.2-1B	ET	0.831	-1.53×10^{-4}	-1.06×10^{-2}
	VB	0.774	-4.13×10^{-5}	-1.35×10^{-2}
	SR	0.494	-1.36×10^{-4}	-7.01×10^{-3}
Llama 3.1-8B	ET	0.639	1.41×10^{-5}	-6.46×10^{-3}
	VB	0.665	-1.02×10^{-4}	-1.48×10^{-3}
	SR	0.119	7.97×10^{-4}	-7.93×10^{-3}
OLMo 2-7B	ET	0.615	-6.72×10^{-5}	-3.25×10^{-4}
	VB	0.573	-1.06×10^{-4}	-1.21×10^{-3}
	SR	0.166	-2.24×10^{-4}	-6.44×10^{-4}
Qwen 3-8B	ET	0.881	1.23×10^{-4}	-1.58×10^{-3}
	VB	0.944	-7.75×10^{-5}	-1.12×10^{-3}
	SR	0.460	2.08×10^{-4}	-2.45×10^{-4}
Qwen 3-14B	ET	0.882	-2.91×10^{-4}	-2.71×10^{-3}
	VB	0.910	9.30×10^{-5}	-6.41×10^{-4}
	SR	0.496	1.09×10^{-4}	-4.91×10^{-4}
Qwen 3-32B	ET	0.831	2.55×10^{-4}	5.15×10^{-4}
	VB	0.922	-1.59×10^{-4}	1.13×10^{-5}
	SR	0.058	-3.84×10^{-4}	-6.33×10^{-4}

Verifying information transfer through group value patching. Prior circuit studies have identified multiple attention heads that jointly contribute to information processing (Olsson et al., 2022; Prakash et al., 2024; Todd et al., 2024; Cho et al., 2026). To assess group effects, we jointly intervene on the functional-DTHs identified above using the same value patching protocol and recovery metric. To test whether this information transfer is specific to functional-DTHs and repeated token positions (anchors), we introduce two controls. One retains the functional-DTHs but replaces information from randomly selected non-anchor positions between the anchor and target (*Functional-DTHs / Non-anchors*). The other retains the anchors but uses random heads matched to the functional group in number and layer distribution (*Random heads / Anchors*). Across all six models and three task families, patching anchor values through functional-DTHs yields higher recovery than either control (Table 1). The magnitude varies across settings, with weaker recovery for the spatial-reasoning task. These results support the direct routing path across the studied models and tasks: contextual information at earlier repeated token positions can pass through functional-DTHs to the target and affect the answer.

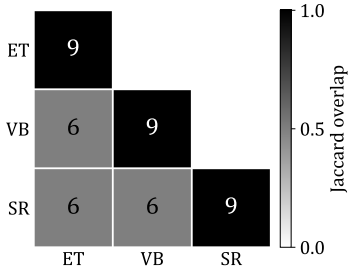

Figure 4: Overlap of functional-DTH.

Functional-DTHs shared across tasks. To examine whether the identified components are task-specific, we compare functional head sets selected independently in the three binding tasks. For a fixed model, we measure the Jaccard overlap between functional-DTH sets D_A and D_B from tasks A and B as $J(D_A, D_B) = |D_A \cap D_B|/|D_A \cup D_B|$. For Llama-3.2-1B, all three sets are identical: L6H8, L6H11, L8H17, and L8H19. Qwen3-8B has nine functional-DTHs per task, six of which are shared by all three tasks (Figure 4). The other models also have subsets shared across the tasks (see Appendix D.3). Together with the value patching effects, this overlap indicates that the same routing components can transfer task-relevant contextual information in different tasks.

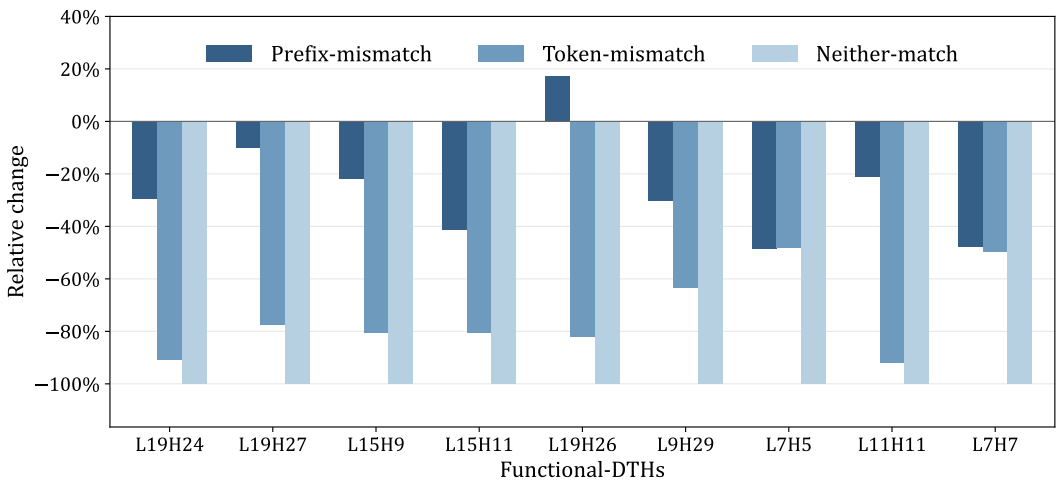

Figure 5: Normalized matching effects of functional-DTHs identified in the colour-container dataset. Both-match is the zero reference; Neither-match is -100%.

3 QK Analyzing for Addressing Mechanism

Having identified functional components and tested their information-transfer effects, we ask how they address the relevant source not sorely rely on the single-token information. Consider a query containing `black bag` when the context contains both `black bag` (we call the `bag` position supplying query-relevant information the source anchor) and `blue bag`. Both anchor positions have the token identity `bag`, so token equality alone cannot distinguish them. Accurately addressing the source associated with the query requires information about the colour prefix.

Controlled token–prefix matching. We use a colour-container variant of entity tracking in which color–container phrases replace letter identifiers (see Appendix A.1 for dataset details). We first identify functional-DTHs within this task using Section 2.3's procedure, then conduct this experiment. For each input, we fix the source anchor and target positions and manipulate their container nouns and colors prefix. For a target phrase `black bag`, the four matching relations are shown in Table 2. Thus, **Prefix-mismatch** preserves token matching but breaks prefix matching; **Token-mismatch** does the reverse. For each functional-DTH, we report relative changes in mean attention logits relative to Both-match, normalized by the logit gap between Both-match and Neither-match. This sets Both-match to 0% and Neither-match to -100%.

Table 2: Token–prefix matching conditions for the target phrase `black bag`.

Source phrase	Condition
`black bag`	Both-match
`blue bag`	Prefix-mismatch
`black case`	Token-mismatch
`blue case`	Neither-match

Prefixes modulate addressing. For most functional-DTHs, Prefix-mismatch leads to a marked decrease in attention logits (Figure 5). Addressing is therefore not merely a token-equality check: contextual information also modulates which repeated token position is favored. This provides a mechanism for distinguishing the source anchor relevant to the query.

Decomposing attention logits into four parts. To locate the query–key interactions underlying prefix modulation, we decompose the residual stream representation after attention-input normalization, $\tilde{r}_{l,p}$ at layer l and position p, into **token identity** (tok) part and **contextual information** (ctx) part. Specifically, we subtract from each token's representation the mean representation of alternative tokens placed in the same neutral context, then average these differences across contexts to estimate its identity direction (see Appendix B.1). Orthogonalizing the span of these directions yields a shared orthonormal basis U_l for each model and layer. With $P_l = U_l U_l^\top$ denoting the projection onto this subspace and I the identity matrix, we obtain $\tilde{r}^{\mathrm{tok}}_{l,p}$ as token identity part and $\tilde{r}^{\mathrm{ctx}}_{l,p}$ as contextual information part. These orthogonal components sum to $\tilde{r}_{l,p}$. All heads and matching conditions within a layer share the same projector.

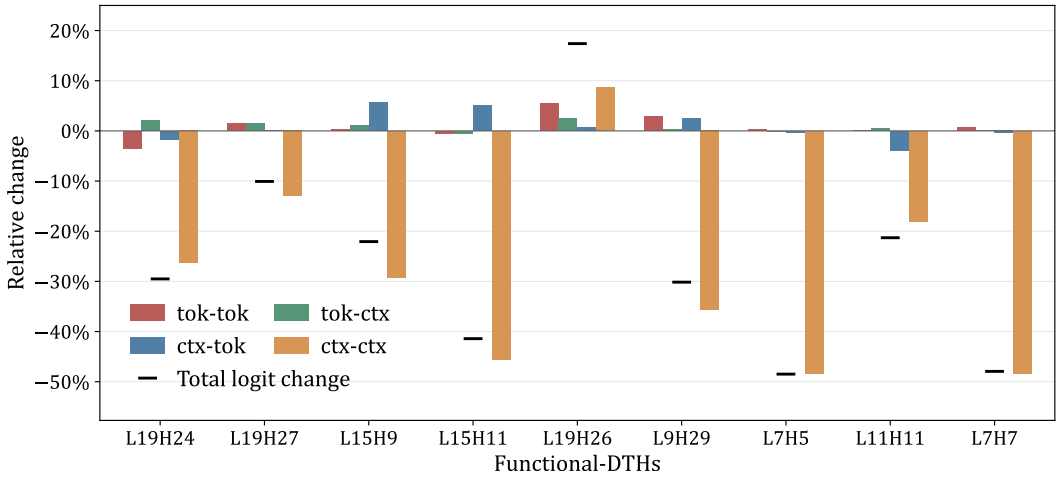

Figure 6: Channel accounting of Prefix-mismatch effects. Colored bars show each channel's mean change from Both-match, normalized by the same full-logit matching advantage as in Figure 5. Black marks show the corresponding total Prefix-mismatch changes.

For a fixed input, layer, and head, we map $\tilde{r}_{l,p}^{\mathrm{tok}}$ and $\tilde{r}_{l,p}^{\mathrm{ctx}}$ through the model's Q/K projections and positional transformations, obtaining additive decompositions of the target (t) query, $\hat{q}_t = \hat{q}_t^{\mathrm{tok}} + \hat{q}_t^{\mathrm{ctx}}$, and anchor ($a$) key, $\hat{k}_a = \hat{k}_a^{\mathrm{tok}} + \hat{k}_a^{\mathrm{ctx}}$. Therefore, the attention logit (ℓ) from t to a can be expanded into for parts:

$$\ell_{t,a} = c_{t,a}^{\mathrm{tok,tok}} + c_{t,a}^{\mathrm{tok,ctx}} + c_{t,a}^{\mathrm{ctx,tok}} + c_{t,a}^{\mathrm{ctx,ctx}}. \tag{1}$$

The four channels correspond to the pairwise interactions between the tok and ctx components of the target query and those of the anchor key. Each term represents the contribution to the attention computation from a different component of the hidden states, namely the token embedding (tok) carried by the residual stream and the contextual information (ctx) introduced through residual branches. By computing these terms separately, we can characterize the addressing preference of an attention head. The decomposition remains exact for models with Q/K normalization because both components use the same fixed normalization factors from the full forward pass (see Appendix B.2).

We then compare each channel's mean value between Prefix-mismatch and Both-match, normalizing the difference by the same full-logit gap. The four normalized channel changes therefore sum to the total Prefix-mismatch effect. Figure 6 shows these changes alongside their total, marked in black. Negative and positive terms contribute to decreases and increases in the attention logit, respectively.

Ctx-ctx interactions dominate Prefix-mismatch effects. Eight of Qwen3-8B's nine functional-DTHs have lower anchor-directed attention logits under Prefix-mismatch. In all eight, ctx-ctx is the largest negative channel term (other models appear in Appendix D.4). For most functional-DTHs, the advantage of prefix matching is thus primarily expressed through interactions between contextual information in the target query and the anchor key. This is consistent with the **previous-token head** mechanism described in induction-head research: these heads write information about the preceding token into the current position, enabling later heads to match using that information (Olsson et al., 2022). Our results support the interpretation that, even when target and anchor tokens match, prefix mismatch can weaken their contextual query–key interaction and lower the attention logit. Prefix information thus provides an additional addressing cue for selecting a relevant source anchor.

4 Evaluating the Causal Role of DTH

The preceding analyses identify direct routing and characterize its addressing mechanism, but do not establish how strongly models rely on it, since competitive mechanism build on causal self-attention can lead the target in query to access earlier positions and potentially reconstruct task-relevant information rather than reuse information already encoded at repeated tokens. An intuitive yet exciting hypothesis is that this competitive mechanism may operate for shorter inputs but break

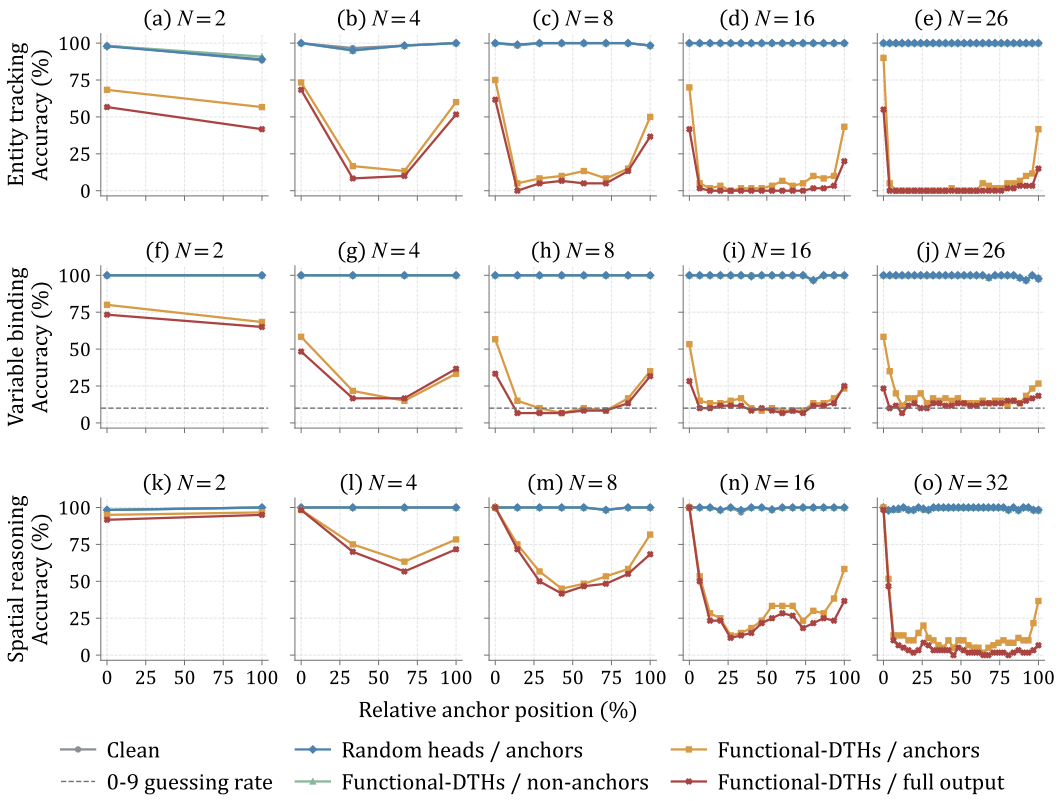

Figure 7: The results of anchor message ablation. Relative anchor position increases toward the query, from 0% (the first of N context items) to 100% (the last item, closest to the query). *Clean* means no ablation. *Random heads / anchors* means removing anchor-to-target messages from layer-matched random heads. *Functional-DTHs / non-anchors* means removing messages from non-anchor positions through functional-DTHs. *Functional-DTHs / anchors* means removing anchor-to-target messages from functional-DTHs. *Functional-DTHs / full output* means zeroing the entire output of functional-DTHs at the target. The dashed *0–9 guessing rate* means uniform guessing among the ten digits (10%), applicable only to variable binding.

down as the input grows longer, thereby shifting control to a DTH-dominated mechanism. To test this, we ablate anchor-to-target messages while varying context length and anchor position.

Anchor message ablation. We adapt the intervention in Section 2 from information replacement to removal: we jointly zero the messages that functional-DTHs write from anchors to the target. Alongside the unmodified Clean condition, we include three controls: removing messages from random non-anchor positions in the same functional-DTHs; removing anchor messages in a group of random heads matched in number and layer distribution; and zeroing the entire target position output of the functional head group. The first two test source position and head specificity. The last tests whether information read from anchors accounts for most of the functional-DTHs' contribution to task performance. All five conditions use the same inputs and are evaluated by full-vocabulary open-ended accuracy, hereafter accuracy.

We vary the number (N) of context items to increase context length and complexity. An item is an object–box relation, a variable assignment, or a spatial relation. We display $N \in \{2, 4, 8, 16, 26\}$ for entity tracking and variable binding, with the final count replaced by $N = 32$ for spatial reasoning (see Appendix A.3 for input construction). For each N, we test the anchor at every position $p \in \{1, \ldots, N\}$ (Figure 7), and find:

The model strongly relies on DTH. As shown in Figure 7, across all the settings, both random-ablation controls closely follow the clean one. Removing only the anchor messages of functional-DTHs, however, substantially reduces accuracy, even for simple inputs containing only four context items, showing that the model relies strongly on the DTH-induced information transmission.

Anchors are the primary information source for direct routing. Ablating only anchor-to-target messages produces a similar accuracy drop to removing the functional-DTHs' entire output at the target, whereas ablating messages from non-anchor positions leaves performance close to clean. Together, these findings suggest that repeated-token anchors are the primary source of reusable task-relevant information for this circuit, highlighting the specificity of direct routing: it selectively addresses repeated token positions to reuse the information they carry.

Reliance increases as contexts become longer and more complex. After anchor-message ablation, we observe two trends for non-initial anchors. At comparable relative positions, increasing the number of context items generally places the anchor farther from the query and reduces accuracy. Similarly, with the number of items held fixed, accuracy tends to recover as the anchor moves closer to the query. Together, these trends suggest stronger reliance on direct routing in longer, more complex contexts.

Together, these phenomena paint a coherent picture: *A competitive mechanism between local information collection and long-range addressing and forwarding.* As prior work on self-repair further shows, other mechanisms can compensate when a computational path is disrupted (Rushing & Nanda, 2024), these results suggest that alternative circuits can still support correct predictions about early items, especially the first, after direct-routing ablation. In contrast, predictions about interior items rely heavily on direct routing, even in simple inputs containing only four items. For shorter inputs, although ablating these heads causes some degradation in output performance, the effect is not substantial. This suggests that even without using the information stored in these anchors, the model can still solve the tasks through an additional local information-collection mechanism. However, as the context becomes longer and the required information is no longer local, the model must rely on the DTH-based long-range information-forwarding mechanism to perform the task, making ablation result in severe accuracy degradation.

Reliance depends on the anchor's position. After anchor-message ablation, performance is generally better for the first item than for interior items. This suggests that the potential competing mechanism may preferentially attend to the beginning and end of the input, whereas DTHs treat input positions more uniformly, which is consistent with positional advantages in long-context retrieval (Liu et al., 2024) and the first-assignment heuristic in variable binding (Wu et al., 2025).

Table 3: Qwen3-8B accuracy (%) on 2,048 pairs per ICL task. Base uses intact demonstrations; Shuffled permutes their answers. Patching transfers values from Base to Shuffled through shared functional-DTHs (DTHs) or layer-matched random heads (Random), reading from demonstration output colons (`A:`), input colons (`Q:`), or non-anchor positions. Parentheses show changes from Shuffled in percentage points; gray highlights anchor patching on generation tasks. FP denotes Financial PhraseBank. Scoring and control details appear in Appendix C.1.

	Task	Base	Shuffled	DTHs `A:`	DTHs `Q:`	DTHs Non-anchor	Random `A:`
Classification tasks	SST-2	92.09	89.50	89.36 (−0.15)	89.16 (−0.34)	89.50 (+0.00)	89.53 (+0.03)
	SST-5	50.98	50.34	50.59 (+0.24)	50.59 (+0.24)	50.48 (+0.14)	50.49 (+0.15)
	AGNews	81.59	71.97	72.61 (+0.63)	72.07 (+0.10)	72.12 (+0.15)	72.26 (+0.29)
	FP	92.72	86.67	86.72 (+0.05)	86.77 (+0.10)	86.58 (−0.09)	86.63 (−0.04)
Generation tasks	Present-past	98.73	37.35	37.74 (+0.39)	37.16 (−0.20)	37.06 (−0.30)	37.07 (−0.28)
	Present-ing	99.07	35.50	36.23 (+0.73)	35.64 (+0.15)	35.66 (+0.17)	35.74 (+0.24)
	English–French	79.74	34.62	38.28 (+3.66)	34.72 (+0.10)	34.90 (+0.28)	34.91 (+0.29)
	Antonym	68.07	36.57	36.96 (+0.39)	36.96 (+0.39)	36.82 (+0.24)	36.74 (+0.17)
	Synonym	52.25	19.29	20.12 (+0.83)	19.29 (+0.00)	19.21 (−0.08)	19.20 (−0.08)

4.1 Direct Routing in In-Context Learning

The preceding experiments characterize the role of functional-DTHs in binding tasks. We next ask whether these heads can transfer useful information in other input settings. We consider in-context learning (ICL), where a model answers a query after observing input–output demonstrations. Our 8-shot prompts use a shared `Q: input` / `A: output` format, so the output colon repeats across demonstrations and at the query.

We shuffle demonstration answers to disrupt their input–output mappings. Using the functional-DTHs shared across the three binding tasks, we apply the same group value-patching protocol to restore the values read from the unshuffled demonstrations' `A:` colons at the query's `A:` colon, while keeping the shuffled prompt's attention weights fixed. We compare accuracy before and after patching, with input-colon, non-anchor, and random-head controls. Appendix C.1 provides the full protocol and cross-model results.

Table 3 shows two broad trends. On classification tasks, anchor-value patching leaves accuracy largely unchanged. Generation tasks are more sensitive to shuffled answers, and patching yields modest improvements, most clearly in English–French translation. Nevertheless, performance remains well below the intact-demonstration baseline. These results suggest that the same functional-DTHs can supply some useful information in ICL, but their effect is much weaker than in the binding tasks.

5 Conclusion and Discussion

Conclusion. Repeated tokens can serve as information anchors through which language models reuse earlier contextualized states. We characterize a direct routing circuit whose effective information transfer is concentrated in a subset of candidate heads, with components shared across tasks. Addressing is not simply a same-token lookup: contextual matching further modulates it, primarily through interactions between contextual information in the query and key. Repeated identity therefore provides an addressing cue, while context helps distinguish which earlier occurrence to read.

Reliance on this path is also position-dependent. Blocking anchor-to-target writes substantially impairs predictions, with information at interior-to-later positions generally depending more on direct routing than information near the beginning. Successful predictions therefore need not always use the same information path. Extending the analysis to ICL (Appendix 4.1) reveals weaker compensatory effects on generation tasks, consistent with prior findings on reusable task-related states. Together, these results support a conditional cache-and-forward mechanism: repeated input structure allows models to reuse contextual information already formed at earlier positions, rather than relying solely on reconstructing it where needed. This provides mechanistic evidence for how information moves through language models.

AI USE STATEMENT

In this work, LLMs are used to help with writing, experiment coding, and visualization of the results.

REFERENCES

Hamidreza Amirzadeh, Afra Alishahi, and Hosein Mohebbi. How language models prioritize contextual grammatical cues? In *Proceedings of the 7th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pp. 315–336, Miami, Florida, US, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.blackboxnlp-1.21. URL https://aclanthology.org/2024.blackboxnlp-1.21/.

Hakaze Cho, Mariko Kato, Yoshihiro Sakai, and Naoya Inoue. Revisiting in-context learning inference circuit in large language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=xizpnYNvQq.

Hakaze Cho, Haolin Yang, Gouki Minegishi, and Naoya Inoue. Mechanism of task-oriented information removal in in-context learning. In *The fourteenth international conference on learning representations*, 2026. URL https://openreview.net/forum?id=VAv1rrPR1A.

Euirim Choi. Goodwiki dataset, sep 2023. URL https://github.com/euirim/goodwiki.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems, 2021. URL https://arxiv.org/abs/2110.14168.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. Word translation without parallel data, 2017. URL https://arxiv.org/abs/1710.04087.

Qin Dai, Benjamin Heinzerling, and Kentaro Inui. Representational analysis of binding in language models, 2024. URL https://arxiv.org/abs/2409.05448.

Jiahai Feng and Jacob Steinhardt. How do language models bind entities in context? In *The Twelfth International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2310.17191.

Aaron Grattafiori et al. The Llama 3 herd of models, 2024. URL https://arxiv.org/abs/2407.21783.

Xiangming Gu, Tianyu Pang, Chao Du, Qian Liu, Fengzhuo Zhang, Cunxiao Du, Ye Wang, and Min Lin. When attention sink emerges in language models: An empirical view. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=78Nn4QJTEN.

Roee Hendel, Mor Geva, and Amir Globerson. In-context learning creates task vectors. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 9318–9333, Singapore, 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-emnlp.624. URL https://aclanthology.org/2023.findings-emnlp.624/.

Xinting Huang, Madhur Panwar, Navin Goyal, and Michael Hahn. InversionView: A general-purpose method for reading information from neural activations. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/f94cfd15db3f16ee7789b6b7e91ec476-Abstract-Conference.html.

Najoung Kim and Sebastian Schuster. Entity tracking in language models. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: Long papers)*, pp. 3835–3855, Toronto, Canada, jul 2023. Association for Computational Linguistics. URL https://aclanthology.org/2023.acl-long.213/.

Yuri Kuratov, Aydar Bulatov, Petr Anokhin, Ivan Rodkin, Dmitry Sorokin, Artyom Sorokin, and Mikhail Burtsev. Babilong: Testing the limits of LLMs with long context reasoning-in-a-haystack, 2024. URL https://arxiv.org/abs/2406.10149.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Prompt repetition improves non-reasoning LLMs. *arXiv preprint arXiv:2512.14982*, 2025. URL https://arxiv.org/abs/2512.14982.

Yuxuan Li, Declan Campbell, Stephanie C. Y. Chan, and Andrew Kyle Lampinen. Just-in-time and distributed task representations in language models, 2025a. URL https://arxiv.org/abs/2509.04466.

Zhuowei Li, Zihao Xu, Ligong Han, Yunhe Gao, Song Wen, Di Liu, Hao Wang, and Dimitris N. Metaxas. Implicit in-context learning. In *The thirteenth international conference on learning representations*, 2025b. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/fc034d186280f55370b6aca7a3285a65-Paper-Conference.pdf.

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

Pekka Malo, Ankur Sinha, Pyry Takala, Pekka Korhonen, and Jyrki Wallenius. Good debt or bad debt: Detecting semantic orientations in economic texts. *arXiv preprint arXiv:1307.5336*, 2013. URL https://arxiv.org/abs/1307.5336v2.

Jack Merullo, Carsten Eickhoff, and Ellie Pavlick. Circuit component reuse across tasks in transformer language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=fpoAYV6Wsk.

Meta. Llama 3.2 model card, 2024. URL https://huggingface.co/meta-llama/Llama-3.2-1B.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. Rethinking the role of demonstrations: What makes in-context learning work? In *Proceedings of the 2022 conference on empirical methods in natural language processing*, pp. 11048–11064. Association for Computational Linguistics, 2022. URL https://aclanthology.org/2022.emnlp-main.759/.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. In-context learning and induction heads. *Transformer Circuits Thread*, 2022. URL https://transformer-circuits.pub/2022/in-context-learning-and-induction-heads/index.html.

Nikhil Prakash, Tamar Rott Shaham, Tal Haklay, Yonatan Belinkov, and David Bau. Fine-tuning enhances existing mechanisms: A case study on entity tracking. In *Proceedings of the 2024 International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=8sKcAWOf2D. arXiv:2402.14811.

Nikhil Prakash, Natalie Shapira, Arnab Sen Sharma, Christoph Riedl, Yonatan Belinkov, Tamar Shaham, David Bau, and Atticus Geiger. Language models use lookbacks to track beliefs. In *International Conference on Learning Representations*, volume 2026, pp. 7657–7694, 2026.

Qwen Team. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Cody Rushing and Neel Nanda. Explorations of self-repair in language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 42836–42855. PMLR, 2024. URL https://proceedings.mlr.press/v235/rushing24a.html.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1631–1642, Seattle, Washington, USA, October 2013. Association for Computational Linguistics. URL https://aclanthology.org/D13-1170/.

Zilu Tang, Qiao Zhao, Gabriel Franco, Derry Tanti Wijaya, Aaron Mueller, Sebastian Schuster, and Najoung Kim. Do language models track entities across state changes? In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=PyJ2YOk3sG.

Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, Nathan Lambert, Dustin Schwenk, Oyvind Tafjord, Taira Anderson, David Atkinson, Faeze Brahman, Christopher Clark, Pradeep Dasigi, Nouha Dziri, Michal Guerquin, Hamish Ivison, Pang Wei Koh, Jiacheng Liu, Saumya Malik, William Merrill, Lester James V. Miranda, Jacob Morrison, Tyler Murray, Crystal Nam, Valentina Pyatkin, Aman Rangapur, Michael Schmitz, Sam Skjonsberg, David Wadden, Christopher Wilhelm, Michael Wilson, Luke Zettlemoyer, Ali Farhadi, Noah A. Smith, and Hannaneh Hajishirzi. 2 OLMo 2 furious, 2024. URL https://arxiv.org/abs/2501.00656.

Eric Todd, Millicent L. Li, Arnab Sen Sharma, Aaron Mueller, Byron C. Wallace, and David Bau. Function vectors in large language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=AwyxtyMwaG. arXiv:2310.15213.

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 *The Eleventh International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2211.00593.

Jason Weston, Antoine Bordes, Sumit Chopra, Alexander M. Rush, Bart van Merriënboer, Armand Joulin, and Tomas Mikolov. Towards AI-complete question answering: A set of prerequisite toy tasks, 2015. URL https://arxiv.org/abs/1502.05698.

Yiwei Wu, Atticus Geiger, and Raphaël Millière. How do transformers learn variable binding in symbolic programs? In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 67284–67299. PMLR, 2025. URL https://proceedings.mlr.press/v267/wu25j.html.

Haolin Yang, Hakaze Cho, and Naoya Inoue. Localizing task recognition and task learning in in-context learning via attention head analysis. In *The fourteenth international conference on learning representations*, 2026. URL https://arxiv.org/abs/2509.24164.

Fred Zhang and Neel Nanda. Towards best practices of activation patching in language models: Metrics and methods. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=Hf17y6u9BC.

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems*, volume 28, pp. 649–657, 2015. URL https://proceedings.neurips.cc/paper/2015/hash/250cf8b51c773f3f8dc8b4be867a9a02-Abstract.html.

APPENDIX

A DATASETS AND EVALUATION

A.1 DATASET EXAMPLES AND DONOR–RECIPIENT PAIR CONSTRUCTION

All binding tasks use the same input structure:

$$\underbrace{\text{Task instruction}}_{\text{optional}} + \text{Binding items} + \text{Query},$$

Table 4 gives the task-specific components.

Table 4: Input components for the binding tasks.

Task	Input components
Entity tracking	**Instruction:** None. **Items:** `... The card is in Box C ...` **Query:** `Box C contains the`
Variable binding	**Instruction:** `Each arrow assigns the value on its left to the variable on its right.` **Items:** `... 9 -> o ...` **Query:** `After applying the assignments, o equals`
Spatial reasoning	**Instruction:** `Answer the question based only on the spatial relations stated in the context. Respond with a single word naming the entity.` **Items:** `... The cross is north of the station. ...` **Query:** `What is north of the station? Answer: the`
Colour-container variant	**Instruction:** `Complete the final statement with the correct object, using only the given bindings.` **Items:** `... the ball is in white box ...` **Query:** `white box contains the`

Entity tracking uses seven object–box bindings with uppercase letter identifiers (Kim & Schuster, 2023; Prakash et al., 2024). Variable binding uses four one-hop assignments from digits 0–9 to distinct lowercase letters. Spatial reasoning uses four facts in the bAbI/QA4 style (Weston et al., 2015; Kuratov et al., 2024), with 36 possible query entities and direct questions about a stated relation. The colour-container variant replaces box letters with combinations of eight colours and four nouns (`bag`, `bin`, `box`, and `case`); each seven-item input contains a same-noun competitor, so the noun alone cannot identify the queried container.

Donor–recipient construction. Binding need not encode an answer's semantic content directly at the queried entity: prior work identifies abstract binding IDs (Feng & Steinhardt, 2024) and order-related binding representations (Dai et al., 2024). We therefore change which item contains the queried identifier, rather than only changing its associated answer while leaving the identifier in place. Starting from a base input, we randomly select another item and exchange its identifier with the queried one, keeping the values, their order, and the query unchanged. The partner can occur anywhere among the other items, not just next to the queried item.

For example, a donor containing `The card is in Box C, the bell is in Box O, ...` becomes a recipient containing `The card is in Box O, the bell is in Box C, ...`. Both end with `Box C contains the`, but the correct answer changes from `card` to `bell`; omitted items are identical. The exchanged units are the box letters in entity tracking, the right-hand variable names in variable binding, and the complete colour–noun phrases in the colour-container variant. Spatial reasoning exchanges the complete relation–entity predicates, such as `north of the station`, rather than the entity words alone.

Value patching aligns the queried identifier across the pair, not its absolute position: the source anchor is the identifier token in each context and the target is its repetition in the query. For colour-container inputs, these are the noun tokens in the queried colour–noun phrase. We evaluate both

directions by exchanging donor and recipient roles. Query identifiers are partitioned into fixed ranking, group-patching, ablation, and QK splits; the colour-container variant partitions query colours while sharing the noun vocabulary. Input construction for the ablation experiments is described separately in Appendix A.3, and ICL inputs in Appendix C.1.

A.2 Value Patching and Recovery

Activation patching tests the causal role of internal model components in a specific behavior by replacing intermediate representations (Wang et al., 2023; Zhang & Nanda, 2024). We use **value patching**, an intervention on value vectors that replaces the information read by heads while keeping the attention pattern fixed (Amirzadeh et al., 2024). We apply this idea to specified anchor-to-target information transfer: we choose an input as the donor, construct its counterfactual as the recipient (see Appendix A.1 for pair construction), and run the two inputs separately. In the head under test, we replace the value read from the recipient anchor with the value from the corresponding donor anchor. The output change thus reflects the effect of replacing anchor information along a fixed reading path. We measure this effect using **pooled normalized recovery**, hereafter **recovery**. Intuitively, recovery measures how much of the donor–recipient answer-preference gap is recovered when only the value vectors are patched, allowing us to identify functional-DTHs that transfer task-relevant information along the tested path (Section 2.3) (Zhang & Nanda, 2024).

For one head, let $z_t^R = \sum_{u \le t} \alpha_{t,u}^R v_u^R$ denote its original recipient output before the output projection, with layer and head indices suppressed. Let a_R and a_D be corresponding recipient and donor anchors. Value patching replaces this output by

$$z_t^P = z_t^R + \alpha_{t,a_R}^R \big(v_{a_D}^D - v_{a_R}^R\big). \tag{2}$$

For multiple anchors, we sum the replacement terms over aligned anchor pairs. The result passes through the head's original output projection. This is a replacement of the source value, not an addition of the donor message on top of the unchanged recipient message. In group patching, every intervened head uses its own cached, unmodified recipient output and attention weights, including at later layers. Earlier interventions therefore do not change the reading pattern of another patched head; the rest of the recipient computation proceeds normally. Donor values are specific to each input pair, not averaged across examples.

Recovery. For directed pair i, fix the donor and recipient answer tokens y_i^D and y_i^R. Define their output-logit difference under a model run x as

$$\Delta_i(x) = \ell^{\text{out}}(y_i^D \mid x) - \ell^{\text{out}}(y_i^R \mid x). \tag{3}$$

With x_i^D, x_i^R, and x_i^P denoting the donor, recipient, and patched-recipient runs, respectively, pooled normalized recovery is

$$\text{Recovery} = \frac{\sum_i \big[\Delta_i(x_i^P) - \Delta_i(x_i^R)\big]}{\sum_i \big[\Delta_i(x_i^D) - \Delta_i(x_i^R)\big]}. \tag{4}$$

The sum includes both directions of every pair. The answer contrast is fixed by the pair, not chosen from the model's highest-scoring alternatives. Recovery is zero at the recipient baseline and one when the aggregate donor preference is restored; it is not clipped to $[0, 1]$. Pooling before division avoids averaging unstable ratios from individual pairs with very small donor–recipient gaps. An effectively zero pooled denominator is reported as undefined. Accuracy is a separate measure: for the single-token binding answers, the highest-logit token over the full vocabulary must match the reference continuation, including its tokenizer-specific word boundary. It is not a forced choice between the two paired answers. Appendix C.1 specifies the separate accuracy evaluation for ICL.

A.3 Input Construction for Anchor Message Ablation

We generate inputs independently of the patching pairs, using the six query identifiers reserved for each task's ablation split. Entity tracking and variable binding use $N \in \{1, 2, 4, 8, 16, 26\}$ items; spatial reasoning uses $N \in \{1, 2, 4, 8, 16, 32\}$. Each input has one queried item and $N - 1$ other items with distinct identifiers, followed by the query. Task instructions and query formats remain those in Table 4.

For each item count N and queried-item position $p \in \{1, \ldots, N\}$, we construct 60 inputs: all combinations of six query identifiers and ten reference answers. Variable binding uses the digits

0–9; entity tracking and spatial reasoning use fixed ten-word answer sets. For a given N, query identifier, and answer, we retain the same other items in the same order and insert the queried item at each possible position. Thus, moving the anchor does not also change the surrounding facts. This yields 3,420 inputs each for entity tracking and variable binding, and 3,780 for spatial reasoning. Plots report relative position $100(p-1)/(N-1)$ for $N > 1$, so 0% and 100% correspond to the first and last items. The single-item condition is evaluated but omitted from the main-text positional curves.

All five conditions use these same inputs and the head group selected on the ranking split, without further head selection. Anchor-message ablation subtracts the original attention-weighted source value from each functional-DTH's target output; it neither removes the input token nor renormalizes attention. Full-output ablation instead sets that head's complete output to zero only at the target. We average the two random controls over ten fixed sets. Non-anchor controls sample the same number of source positions strictly between the anchor and target, excluding special tokens and other occurrences of the target token; positions are reused only if fewer than ten choices exist. Random-head groups match the functional group in size and layer distribution and exclude the functional-DTHs, but need not exclude other DTH candidates. Accuracy is averaged over the 60 inputs in each (N, p) cell.

B Representation Decomposition

B.1 Estimating Token Identity

We estimate identity from neutral text independently of the task inputs. The vocabulary comprises 14 container nouns: `bag`, `basket`, `bin`, `bowl`, `box`, `can`, `case`, `chest`, `crate`, `cup`, `drawer`, `jar`, `tin`, and `tray`. It includes all four nouns manipulated in the QK experiment. Each noun is inserted into the same 200 frozen contexts, producing 2,800 neutral inputs per model. For example, `Before the inspection, the [noun] had not attracted much attention.` is instantiated separately for every noun. At the noun position, we record the activation actually supplied to the Q/K projections, after attention-input normalization. Under causal attention, only the text up to that position affects this activation.

Let $\tilde{r}_{l,c}(w)$ denote this representation for noun w in context c at layer l, and let $\mathcal{V}$ be the 14-word vocabulary. For each context, the other 13 nouns form a matched background. We first subtract that background, then average across contexts:

$$b_{l,c}(w) = \frac{1}{13} \sum_{u \in \mathcal{V} \setminus \{w\}} \tilde{r}_{l,c}(u), \qquad \mu_l(w) = \frac{1}{200} \sum_{c=1}^{200} \big[\tilde{r}_{l,c}(w) - b_{l,c}(w)\big]. \tag{5}$$

The background is therefore obtained from alternative nouns in separate, otherwise identical inputs, not from neighboring token positions in one sentence.

We normalize each $\mu_l(w)$ to unit length, stack these directions as columns, and use the left singular vectors spanning their column space as U_l. Singular values below $\max(10^{-6}, 10^{-4}\sigma_{\max})$ are discarded; the resulting basis has rank 13 in all analyzed model–layer combinations. As in Equation **??**, $P_l = U_l U_l^\top$ defines token identity and $I - P_l$ defines contextual information. The basis is model- and layer-specific, but shared by every head and matching condition within that layer. It is estimated once and cached; task performance and matching effects are not used to choose its directions.

B.2 Exact Four-Channel Decomposition with Q/K Normalization

For a fixed input and layer, the Q/K projections are linear in the attention-input activation. The evaluated models have no nonzero Q/K projection bias, so $u = W\tilde{r} = u^{\mathrm{tok}} + u^{\mathrm{ctx}}$, where W denotes either projection. Q/K normalization is absent in the two Llama models, is applied within each head in Qwen3, and is applied to the full query or key projection before splitting heads in OLMo 2. The calculation below respects each model's normalization scope.

For an RMS-normalized vector u, let d be the width over which the model computes its RMS and g its learned gain. Conditional on the full forward-pass vector, define

$$\rho(u) = \sqrt{\frac{1}{d}\sum_{j=1}^{d} u_j^2 + \epsilon}, \qquad M(u) = \frac{\mathrm{diag}(g)}{\rho(u)}. \tag{6}$$

Although normalization is nonlinear as a function of u, $M(u)$ is a fixed linear map when decomposing this particular forward pass. We apply it to both components without recomputing their RMS values:

$$\mathrm{RMSNorm}(u) = M(u)u^{\mathrm{tok}} + M(u)u^{\mathrm{ctx}}. \tag{7}$$

Without Q/K normalization, we simply use the identity map. Head extraction and rotary positional transformations are also linear. Applying the original query and key paths to each component therefore gives $\hat{q}_t = \hat{q}_t^{\mathrm{tok}} + \hat{q}_t^{\mathrm{ctx}}$ and $\hat{k}_a = \hat{k}_a^{\mathrm{tok}} + \hat{k}_a^{\mathrm{ctx}}$. With $\gamma, \eta \in \{\mathrm{tok}, \mathrm{ctx}\}$,

$$c_{t,a}^{\gamma,\eta} = \frac{(\hat{q}_t^{\gamma})^{\top}\hat{k}_a^{\eta}}{\sqrt{d_h}}, \qquad \ell_{t,a} = \sum_{\gamma,\eta} c_{t,a}^{\gamma,\eta}. \tag{8}$$

For grouped-query attention, the key is taken from the KV head assigned to the query head. Each matching condition uses its own full forward pass and its own normalization factors. These are additive terms of the actual attention logit, not counterfactual scores obtained by independently normalizing an isolated tok or ctx component. Softmax attention weights are not decomposed additively.

The equality holds in real arithmetic. In the implementation, native forward activations are BF16, while component projections and dot products are accumulated in FP32. Finite-precision discrepancies between the channel sum and the native attention logit are recorded without filtering heads or inputs, or forcing the sum to match by reallocating the residual error.

C In-Context Learning Experiments

C.1 Input Construction and Additional Results

This section supplements the ICL experiments in Section 4.1 with input construction, scoring details, and complete cross-model results.

Motivation. Task and function vectors extracted from few-shot inputs can improve zero-shot performance (Hendel et al., 2023; Todd et al., 2024). More detailed positional analyses locate some transferable task states at the colons preceding demonstration answers (Li et al., 2025a). Removing format information such as `A:` when constructing context vectors can also reduce their effectiveness (Li et al., 2025b). These findings suggest that formatting positions can carry reusable states. Meanwhile, ICL circuit studies identify paths that read demonstration labels and components that use the query's own semantics (Cho et al., 2025; Yang et al., 2026). We therefore ask whether restoring information supplied through repeated colons by the previously identified functional-DTHs can compensate for disrupted demonstration input–output mappings.

Task data, prompt construction, and scoring. We use SST-2 and SST-5 (Socher et al., 2013), AGNews (Zhang et al., 2015), Financial PhraseBank (FP) (Malo et al., 2013), Present-past and Present-ing transformations, English–French translation, and antonym and synonym prediction (Hendel et al., 2023; Todd et al., 2024; Conneau et al., 2017). All nine tasks use the same 8-shot Q/A format without an additional task instruction. SST-2 uses `negative` and `positive`; SST-5 additionally distinguishes `very negative`, `neutral`, and `very positive`; AGNews uses `world`, `sports`, `business`, and `science and technology`. FP uses the all-agree subset, with `negative`, `neutral`, and `positive` labels; its 2,264 source sentences yield 2,259 unique eligible inputs after deduplication.

The following table lists actual query inputs and their dataset reference answers. These rows illustrate individual mappings, not complete zero-shot prompts: each query is preceded by eight demonstrations in the experiment.

Task	Query input	Reference answer
SST-2	`the diverse , marvelously twisted shapes history`	`positive`
SST-5	`We never really feel involved with the story , as all of its ideas remain just that : abstract ideas .`	`very negative`
AGNews	`Flashbacks and Financial Lessons Fleeting childhood moments can have profound financial consequences.`	`business`
FP	`Talentum expects that the net sales of its core business will increase in 2008 , compared to 2007 .`	`positive`
Present-past	`wash`	`washed`
Present-ing	`wash`	`washing`
English–French	`childhood`	`enfance`
Antonym	`complainant`	`defendant`
Synonym	`very`	`extremely`

Each task contains 2,048 distinct ordered base prompts. For each prompt, we randomly draw nine distinct eligible inputs: eight form the demonstrations and the ninth is the query. Inputs may recur or switch roles across prompts, so 2,048 pairs do not imply 2,048 unique queries. We independently sample one answer permutation per base, rejecting unchanged answer sequences; the realized pairs are then fixed across models and interventions. Present-past and Present-ing share query verbs, demonstration verbs, and answer permutations. We retain the dataset reference answers, including multi-token answers; these examples are not model-generated responses. For each model and task, we use full-vocabulary next-token accuracy if every reference continuation is single-token, and complete generated-answer exact match otherwise. Antonym and Synonym retain reference-first-token accuracy following the Function Vectors protocol (Todd et al., 2024); alternative synonyms or antonyms are not added to the reference set.

A complete Present-past example is shown in Table 5. Each demonstration places `Q: input` and `A: output` on separate lines, and a blank line separates consecutive demonstrations and the final query. The final `A:` has no trailing space.

The shuffled input retains every demonstration input and the final query, but permutes complete demonstration answers. Its reference answer remains `washed`. The eight demonstration `A:` colons are the source anchors; the final query's `A:` colon is the target. The `Q:` colons are distinct control positions. Patching therefore transfers values from the intact prompt's output-marker positions, not from the answer words that follow them. Alignment is by demonstration slot, since answer permutations can change absolute token positions.

Intervention and controls. We use each model's functional-DTHs shared across the three binding tasks in Section 2.3, without selecting heads anew on ICL. The intact input is the donor and its shuffled counterpart is the recipient. Following Appendix A.2, we jointly replace the values read from the eight aligned demonstration `A:` colons, keeping each intervened layer's original recipient attention weights and writes from other sources fixed. This is value replacement, not an additive vector intervention. Under causal attention, an anchor can encode earlier demonstrations and its current input but cannot read its own following answer.

The controls use the eight demonstration `Q:` colons, eight aligned random non-anchor positions, or the original `A:` anchors through random heads matched in number and layer distribution. Non-anchor and random-head results average ten fixed control sets. Qwen3-8B uses six shared functional-DTHs; the other models' group sizes are given in their result tables.

Interpretation. The reported changes are point estimates on fixed prompt pairs, not evidence of statistical significance. Shuffling preserves answer-set cues such as label meanings, language, and morphology, so lower accuracy does not establish loss of task recognition (Min et al., 2022; Yang et al., 2026). Donor anchors also contain information from earlier intact demonstrations. The experiment therefore tests the compensatory role of the shared functional-DTHs, without isolating abstract task identity from demonstration content or uniquely identifying the transferred information as a task/function vector.

Table 5: A complete 8-shot Present-past input pair. The shuffled input permutes demonstration answers while preserving all inputs and the query. The reference answer is shown separately and is not part of either prompt.

Base	Shuffled
`Q: stipulate` `A: stipulated`	`Q: stipulate` `A: asked`
`Q: cheat` `A: cheated`	`Q: cheat` `A: stipulated`
`Q: ask` `A: asked`	`Q: ask` `A: cheated`
`Q: wear` `A: wore`	`Q: wear` `A: wore`
`Q: gild` `A: gilded`	`Q: gild` `A: confessed`
`Q: fetch` `A: fetched`	`Q: fetch` `A: fetched`
`Q: enunciate` `A: enunciated`	`Q: enunciate` `A: gilded`
`Q: confess` `A: confessed`	`Q: confess` `A: enunciated`
`Q: wash` `A:`	`Q: wash` `A:`
Reference answer: `washed`	

Cross-model results. The following tables collect the expanded 2,048-pair results for all six models and nine tasks. Earlier 512-pair results are not mixed into these tables. All models use the same realized prompt pairs, with model-specific functional head groups selected from the three binding tasks.

Table 6: ICL accuracy (%) for Llama-3.2-1B with 4 shared functional-DTHs. Conditions, scoring, and parentheses follow Table 3. Each reported row uses all 2,048 prompt pairs; – denotes an incomplete three-hour job, whose partial results are not reported.

	Task	Base	Shuffled	DTHs `A:`	DTHs `Q:`	DTHs Non-anchor	Random `A:`
Classification tasks	SST-2	81.64	73.63	73.14 (−0.49)	73.54 (−0.10)	73.43 (−0.21)	73.53 (−0.10)
	SST-5	39.50	35.84	36.57 (+0.73)	35.89 (+0.05)	35.81 (−0.03)	35.80 (−0.04)
	AGNews	74.12	65.53	65.19 (−0.34)	65.38 (−0.15)	65.44 (−0.08)	65.33 (−0.20)
	FP	67.19	61.43	61.62 (+0.20)	61.87 (+0.44)	61.58 (+0.16)	61.62 (+0.19)
Generation tasks	Present-past	94.82	20.70	22.56 (+1.86)	20.61 (−0.10)	20.73 (+0.02)	20.58 (−0.13)
	Present-ing	96.58	19.24	21.48 (+2.25)	19.19 (−0.05)	19.38 (+0.14)	19.17 (−0.06)
	English–French	71.04	25.59	26.81 (+1.22)	25.39 (−0.20)	25.29 (−0.30)	25.22 (−0.36)
	Antonym	60.35	17.43	18.70 (+1.27)	17.09 (−0.34)	17.39 (−0.04)	17.33 (−0.10)
	Synonym	35.64	7.37	8.06 (+0.68)	7.47 (+0.10)	7.60 (+0.22)	7.56 (+0.19)

Table 7: ICL accuracy (%) for Llama-3.1-8B with 2 shared functional-DTHs. Conditions, scoring, and parentheses follow Table 3. Each reported row uses all 2,048 prompt pairs; – denotes an incomplete three-hour job, whose partial results are not reported.

	Task	Base	Shuffled	DTHs `A:`	DTHs `Q:`	DTHs Non-anchor	Random `A:`
Classification tasks	SST-2	88.33	71.14	71.53 (+0.39)	71.24 (+0.10)	71.41 (+0.27)	71.43 (+0.29)
	SST-5	50.15	43.55	43.99 (+0.44)	43.90 (+0.34)	43.52 (−0.03)	43.59 (+0.04)
	AGNews	82.42	52.10	52.34 (+0.24)	51.95 (−0.15)	51.89 (−0.21)	51.90 (−0.20)
	FP	66.80	47.95	47.61 (−0.34)	48.34 (+0.39)	48.18 (+0.23)	48.04 (+0.09)
Generation tasks	Present-past	99.07	27.39	27.59 (+0.20)	27.39 (+0.00)	27.48 (+0.08)	27.35 (−0.04)
	Present-ing	99.51	28.76	29.30 (+0.54)	28.37 (−0.39)	28.65 (−0.11)	28.56 (−0.20)
	English–French	81.49	12.55	13.33 (+0.78)	12.79 (+0.24)	12.71 (+0.16)	12.62 (+0.07)
	Antonym	70.02	27.69	28.32 (+0.63)	27.69 (+0.00)	27.66 (−0.03)	27.59 (−0.10)
	Synonym	46.39	8.59	8.40 (−0.20)	8.35 (−0.24)	8.38 (−0.21)	8.38 (−0.21)

Table 8: ICL accuracy (%) for OLMo-2-7B with 6 shared functional-DTHs. Conditions, scoring, and parentheses follow Table 3. Each reported row uses all 2,048 prompt pairs; – denotes an incomplete three-hour job, whose partial results are not reported.

	Task	Base	Shuffled	DTHs `A:`	DTHs `Q:`	DTHs Non-anchor	Random `A:`
Classification tasks	SST-2	92.09	91.46	91.41 (−0.05)	91.46 (+0.00)	91.42 (−0.03)	91.42 (−0.03)
	SST-5	50.20	49.66	49.76 (+0.10)	49.80 (+0.15)	49.74 (+0.08)	49.74 (+0.08)
	AGNews	88.92	87.21	87.30 (+0.10)	87.30 (+0.10)	87.35 (+0.15)	87.30 (+0.10)
	FP	47.51	45.02	45.17 (+0.15)	44.92 (−0.10)	45.02 (+0.00)	45.01 (−0.00)
Generation tasks	Present-past	98.93	16.75	16.89 (+0.15)	16.75 (+0.00)	16.76 (+0.01)	16.88 (+0.13)
	Present-ing	99.32	17.38	17.58 (+0.20)	17.43 (+0.05)	17.52 (+0.14)	17.61 (+0.22)
	English–French	77.83	19.19	19.24 (+0.05)	19.34 (+0.15)	19.32 (+0.13)	19.28 (+0.09)
	Antonym	70.80	38.33	38.53 (+0.20)	38.38 (+0.05)	38.26 (−0.07)	38.35 (+0.02)
	Synonym	48.78	21.14	21.04 (−0.10)	21.14 (+0.00)	21.09 (−0.05)	21.17 (+0.03)

Table 9: ICL accuracy (%) for Qwen3-8B with 6 shared functional-DTHs. Conditions, scoring, and parentheses follow Table 3. Each reported row uses all 2,048 prompt pairs; – denotes an incomplete three-hour job, whose partial results are not reported.

	Task	Base	Shuffled	DTHs `A:`	DTHs `Q:`	DTHs Non-anchor	Random `A:`
Classification tasks	SST-2	92.09	89.50	89.36 (−0.15)	89.16 (−0.34)	89.50 (+0.00)	89.53 (+0.03)
	SST-5	50.98	50.34	50.59 (+0.24)	50.59 (+0.24)	50.48 (+0.14)	50.49 (+0.15)
	AGNews	81.59	71.97	72.61 (+0.63)	72.07 (+0.10)	72.12 (+0.15)	72.26 (+0.29)
	FP	92.72	86.67	86.72 (+0.05)	86.77 (+0.10)	86.58 (−0.09)	86.63 (−0.04)
Generation tasks	Present-past	98.73	37.35	37.74 (+0.39)	37.16 (−0.20)	37.06 (−0.30)	37.07 (−0.28)
	Present-ing	99.07	35.50	36.23 (+0.73)	35.64 (+0.15)	35.66 (+0.17)	35.74 (+0.24)
	English–French	79.74	34.62	38.28 (+3.66)	34.72 (+0.10)	34.90 (+0.28)	34.91 (+0.29)
	Antonym	68.07	36.57	36.96 (+0.39)	36.96 (+0.39)	36.82 (+0.24)	36.74 (+0.17)
	Synonym	52.25	19.29	20.12 (+0.83)	19.29 (+0.00)	19.21 (−0.08)	19.20 (−0.08)

Table 10: ICL accuracy (%) for Qwen3-14B with 9 shared functional-DTHs. Conditions, scoring, and parentheses follow Table 3. Each reported row uses all 2,048 prompt pairs; – denotes an incomplete three-hour job, whose partial results are not reported.

	Task	Base	Shuffled	DTHs `A:`	DTHs `Q:`	DTHs Non-anchor	Random `A:`
Classification tasks	SST-2	92.29	90.58	90.48 (−0.10)	90.62 (+0.05)	90.69 (+0.11)	90.66 (+0.09)
	SST-5	–	–	–	–	–	–
	AGNews	–	–	–	–	–	–
	FP	93.16	88.57	88.62 (+0.05)	88.72 (+0.15)	88.74 (+0.17)	88.69 (+0.11)
Generation tasks	Present-past	–	–	–	–	–	–
	Present-ing	–	–	–	–	–	–
	English–French	–	–	–	–	–	–
	Antonym	67.43	35.21	35.94 (+0.73)	35.40 (+0.20)	35.36 (+0.15)	35.41 (+0.21)
	Synonym	51.51	19.04	19.48 (+0.44)	19.09 (+0.05)	19.05 (+0.01)	19.05 (+0.00)

Table 11: ICL accuracy (%) for Qwen3-32B with 7 shared functional-DTHs. Conditions, scoring, and parentheses follow Table 3. Each reported row uses all 2,048 prompt pairs; – denotes an incomplete three-hour job, whose partial results are not reported.

	Task	Base	Shuffled	DTHs `A:`	DTHs `Q:`	DTHs Non-anchor	Random `A:`
Classification tasks	SST-2	92.63	86.62	86.62 (+0.00)	86.52 (−0.10)	86.70 (+0.08)	86.75 (+0.13)
	SST-5	–	–	–	–	–	–
	AGNews	–	–	–	–	–	–
	FP	90.77	80.08	80.08 (+0.00)	80.22 (+0.15)	80.28 (+0.21)	80.21 (+0.14)
Generation tasks	Present-past	–	–	–	–	–	–
	Present-ing	–	–	–	–	–	–
	English–French	–	–	–	–	–	–
	Antonym	68.60	24.61	24.76 (+0.15)	24.61 (+0.00)	24.50 (−0.11)	24.59 (−0.01)
	Synonym	49.32	15.23	15.19 (−0.05)	15.19 (−0.05)	15.10 (−0.14)	15.20 (−0.03)

D ADDITIONAL RESULTS

D.1 DTH-SCORE RANKINGS

Figure 8 reports the complete discovery rankings used to define the candidate pools. The cutoff is fixed before task-specific value patching; it is not tuned separately for each task.

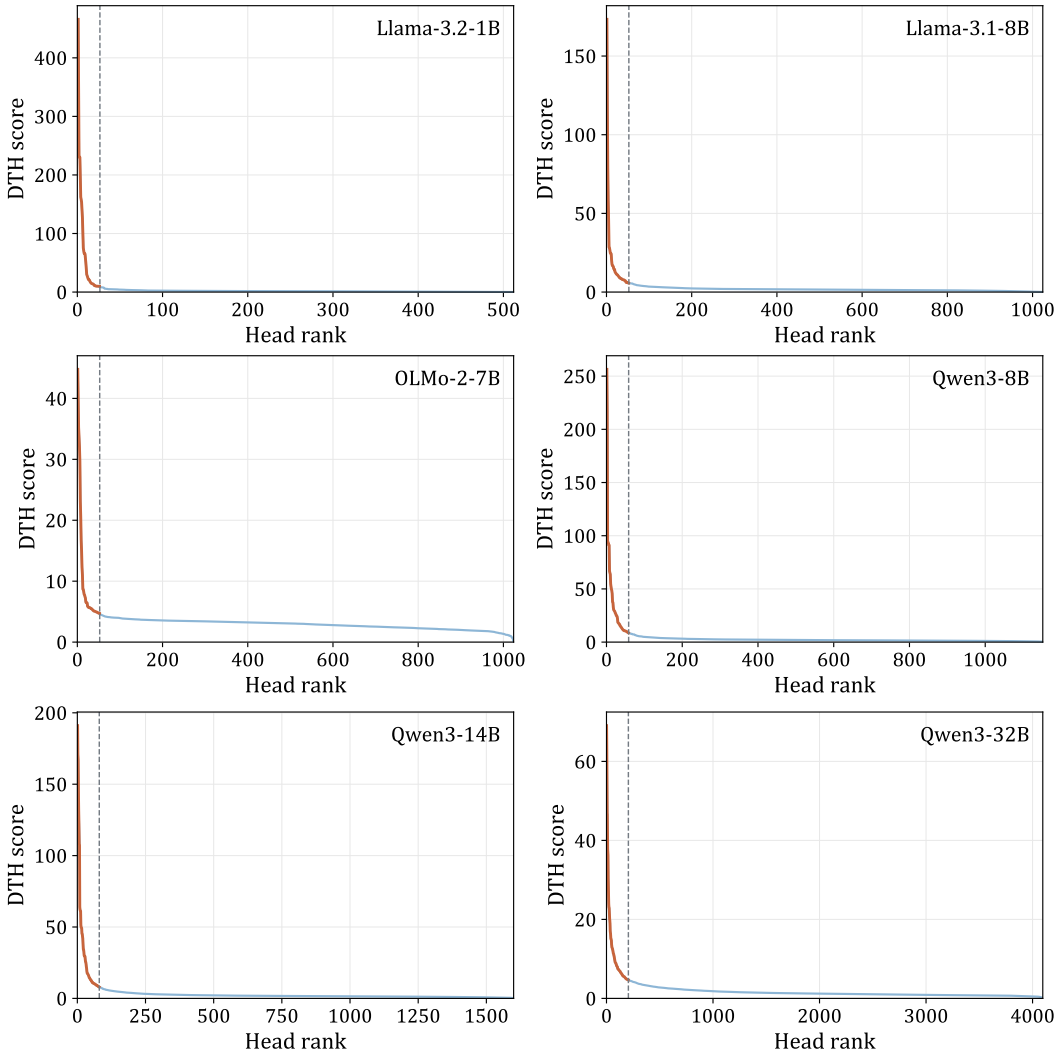

Figure 8: DTH-score rankings on the shared discovery dataset. Heads are ordered by decreasing score; orange denotes the top 5% (rounded up), and dashed lines mark the candidate cutoff. Candidate/total head counts are Llama-3.2-1B: 26/512; Llama-3.1-8B: 52/1024; OLMo-2-7B: 52/1024; Qwen3-8B: 58/1152; Qwen3-14B: 80/1600; Qwen3-32B: 205/4096.

D.2 SINGLE-HEAD VALUE-PATCHING RESULTS

The following plots give the complete, unfolded results corresponding to Figure 3, including the colour-container variant used for QK analysis. Each bar is one candidate's pooled recovery on the ranking split, evaluated alone. The top 15% (rounded up) define the functional group; the dashed line summarizes the layer-matched random heads. Head labels are zero-indexed. Dense rankings are split into consecutive panels without omitting heads or changing the vertical scale within a model–task setting. Selection is based only on these single-head results, not on group-patching or ablation outcomes.

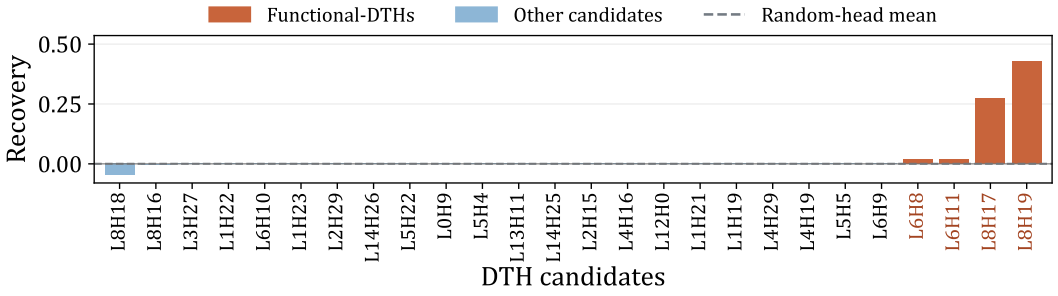

Figure 9: Llama-3.2-1B, entity tracking: 592 pairs (both directions); all 26 candidates.

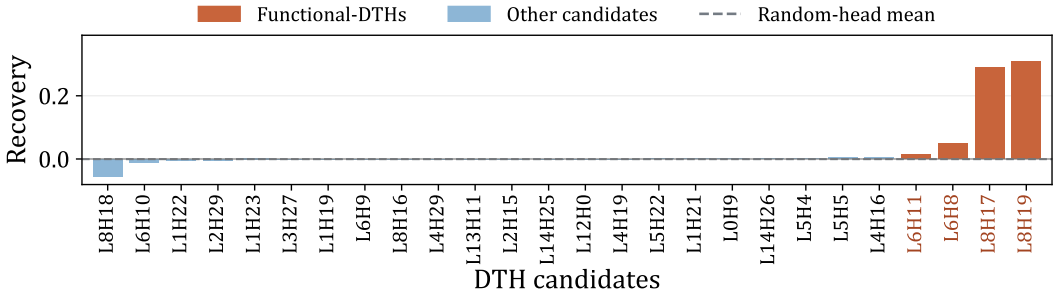

Figure 10: Llama-3.2-1B, variable binding: 350 pairs (both directions); all 26 candidates.

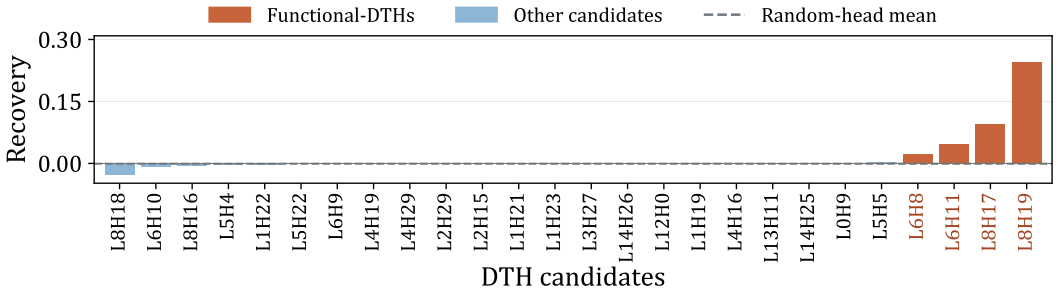

Figure 11: Llama-3.2-1B, spatial reasoning: 576 pairs (both directions); all 26 candidates.

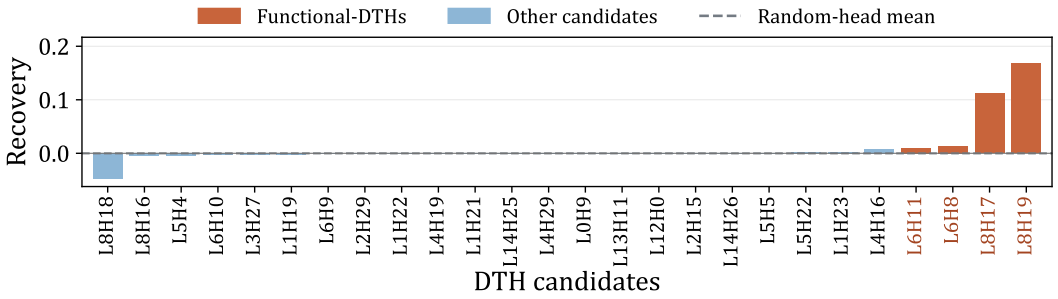

Figure 12: Llama-3.2-1B, colour-container: 560 pairs (both directions); all 26 candidates.

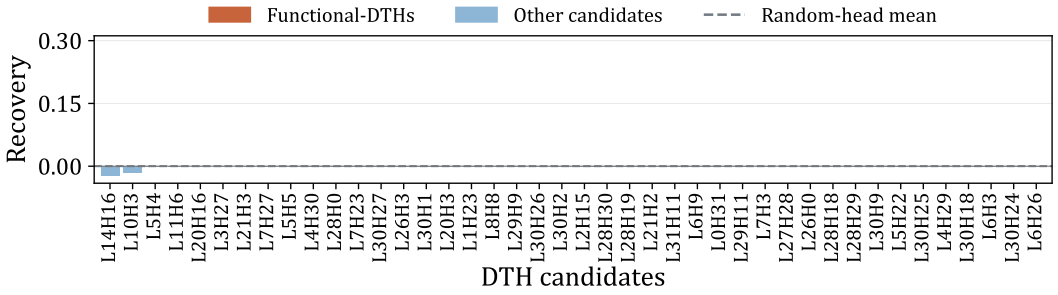

Figure 13: Llama-3.1-8B, entity tracking: 592 pairs (both directions); ranks 1–42/52.

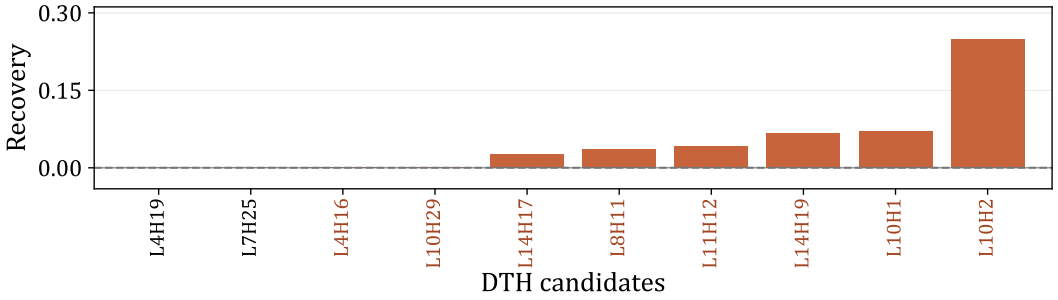

Figure 14: Llama-3.1-8B, entity tracking: 592 pairs (both directions); ranks 43–52/52.

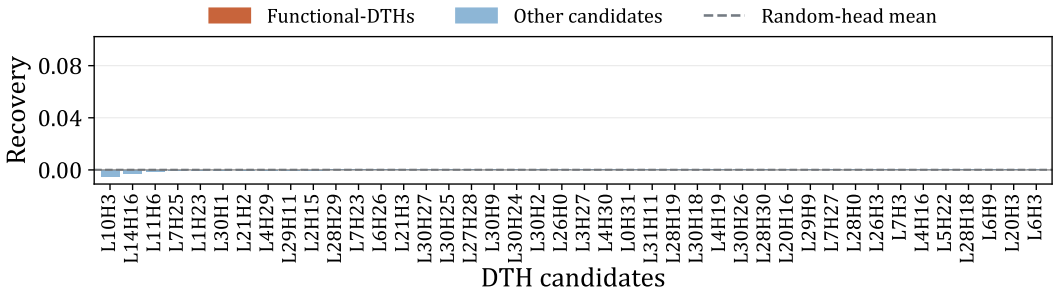

Figure 15: Llama-3.1-8B, variable binding: 350 pairs (both directions); ranks 1–42/52.

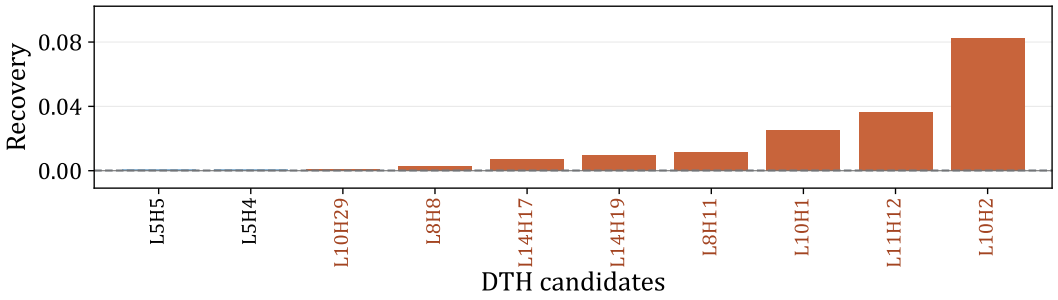

Figure 16: Llama-3.1-8B, variable binding: 350 pairs (both directions); ranks 43–52/52.

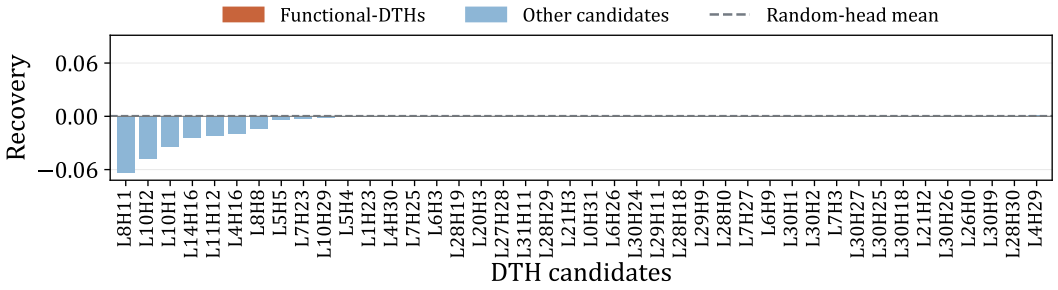

Figure 17: Llama-3.1-8B, spatial reasoning: 576 pairs (both directions); ranks 1–42/52.

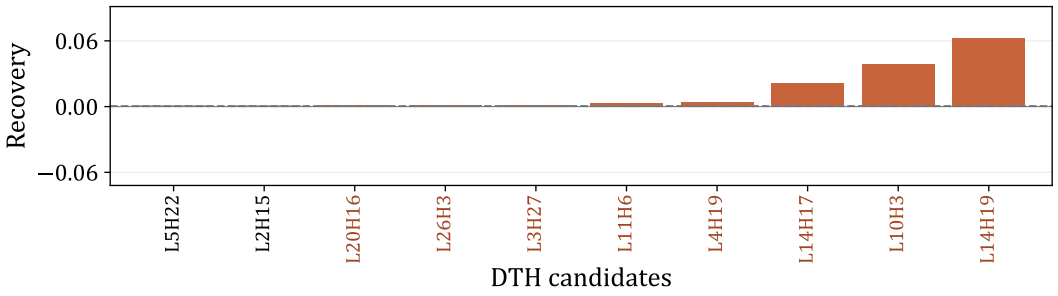

Figure 18: Llama-3.1-8B, spatial reasoning: 576 pairs (both directions); ranks 43–52/52.

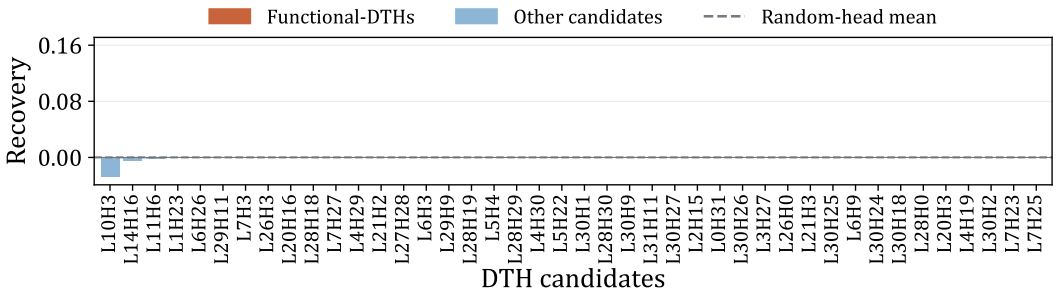

Figure 19: Llama-3.1-8B, colour-container: 560 pairs (both directions); ranks 1–42/52.

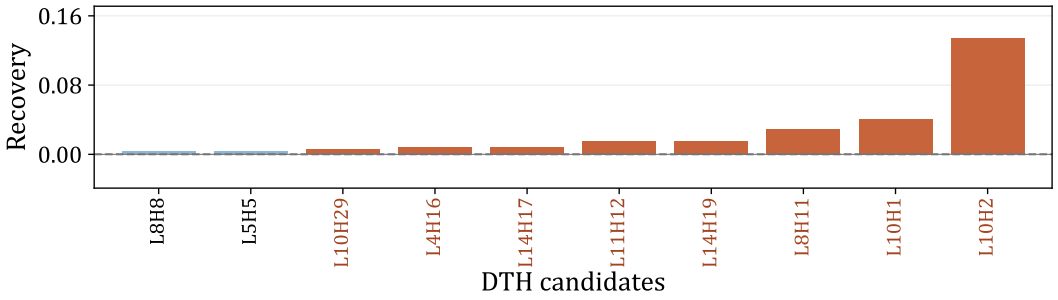

Figure 20: Llama-3.1-8B, colour-container: 560 pairs (both directions); ranks 43–52/52.

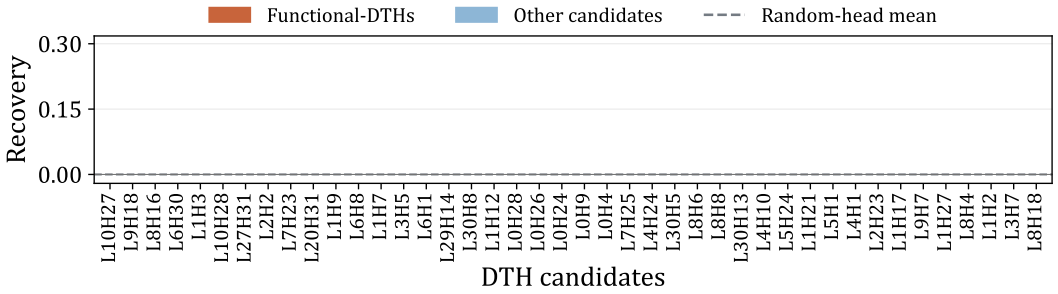

Figure 21: OLMo-2-7B, entity tracking: 592 pairs (both directions); ranks 1–42/52.

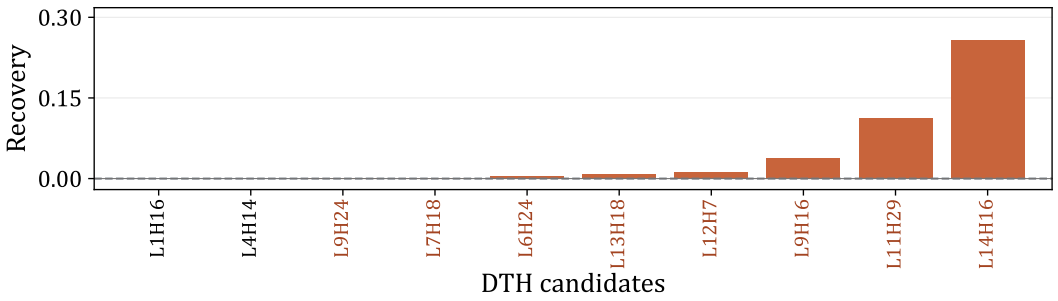

Figure 22: OLMo-2-7B, entity tracking: 592 pairs (both directions); ranks 43–52/52.

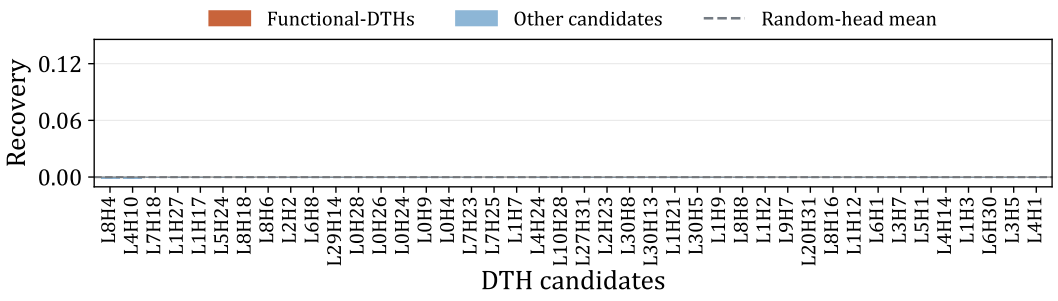

Figure 23: OLMo-2-7B, variable binding: 350 pairs (both directions); ranks 1–42/52.

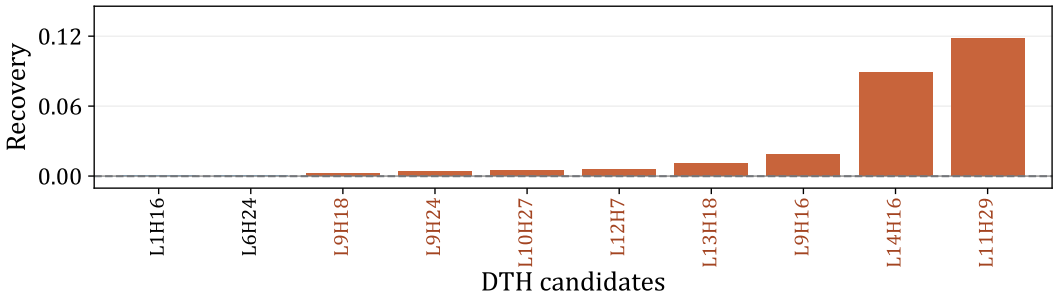

Figure 24: OLMo-2-7B, variable binding: 350 pairs (both directions); ranks 43–52/52.

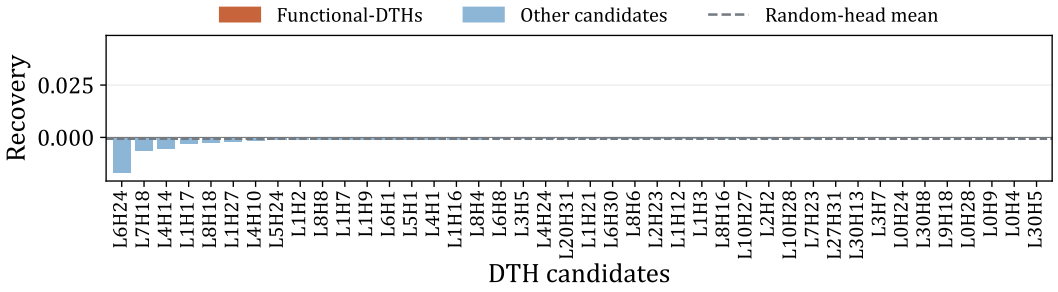

Figure 25: OLMo-2-7B, spatial reasoning: 576 pairs (both directions); ranks 1–42/52.

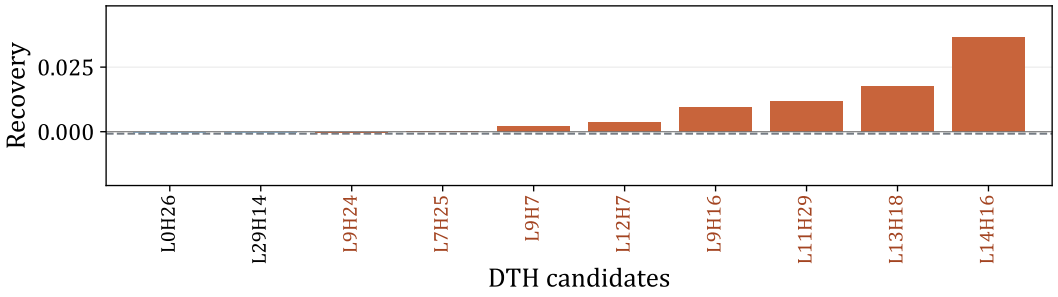

Figure 26: OLMo-2-7B, spatial reasoning: 576 pairs (both directions); ranks 43–52/52.

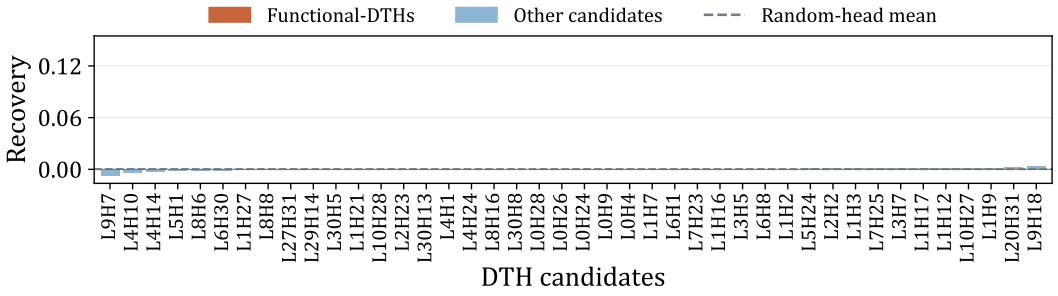

Figure 27: OLMo-2-7B, colour-container: 560 pairs (both directions); ranks 1–42/52.

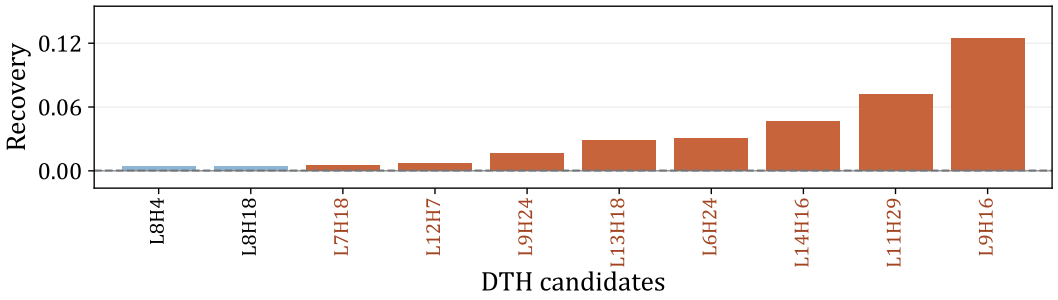

Figure 28: OLMo-2-7B, colour-container: 560 pairs (both directions); ranks 43–52/52.

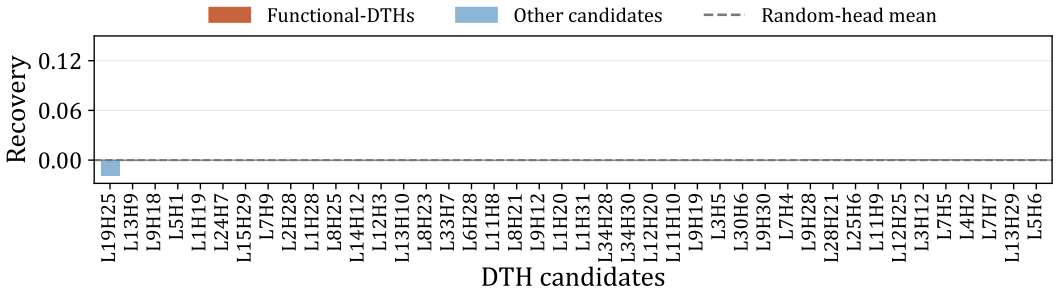

Figure 29: Qwen3-8B, entity tracking: 592 pairs (both directions); ranks 1–42/58.

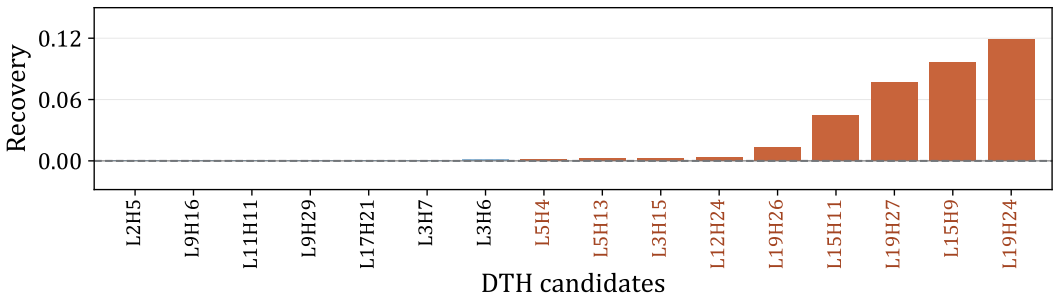

Figure 30: Qwen3-8B, entity tracking: 592 pairs (both directions); ranks 43–58/58.

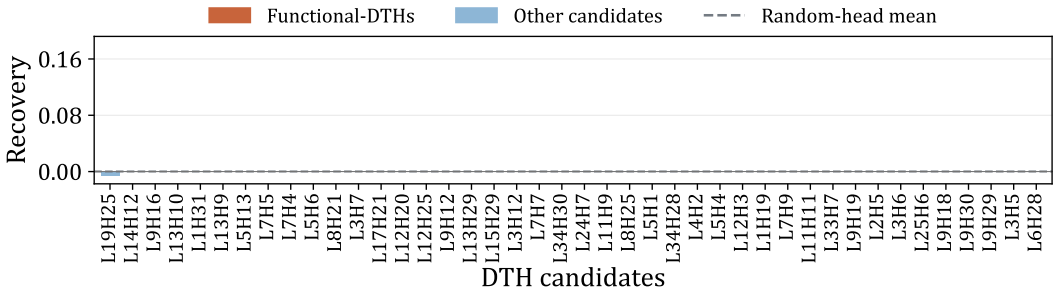

Figure 31: Qwen3-8B, variable binding: 350 pairs (both directions); ranks 1–42/58.

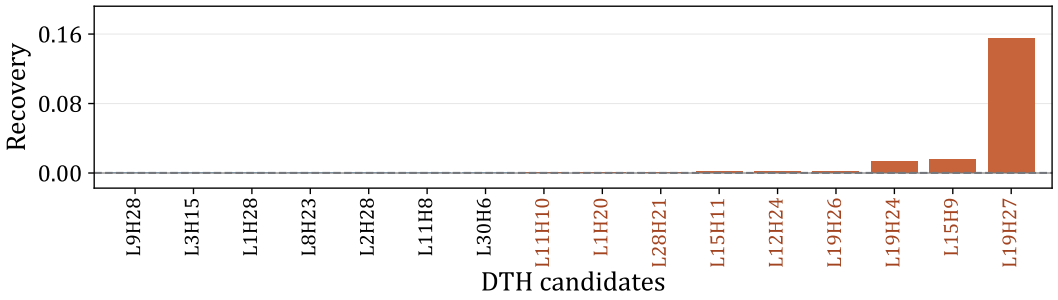

Figure 32: Qwen3-8B, variable binding: 350 pairs (both directions); ranks 43–58/58.

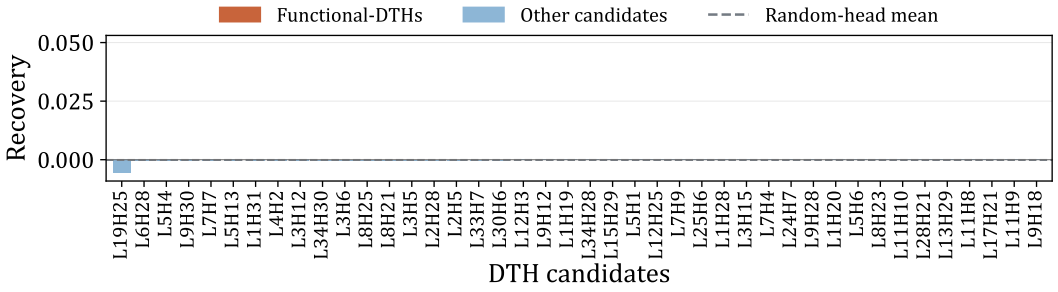

Figure 33: Qwen3-8B, spatial reasoning: 576 pairs (both directions); ranks 1–42/58.

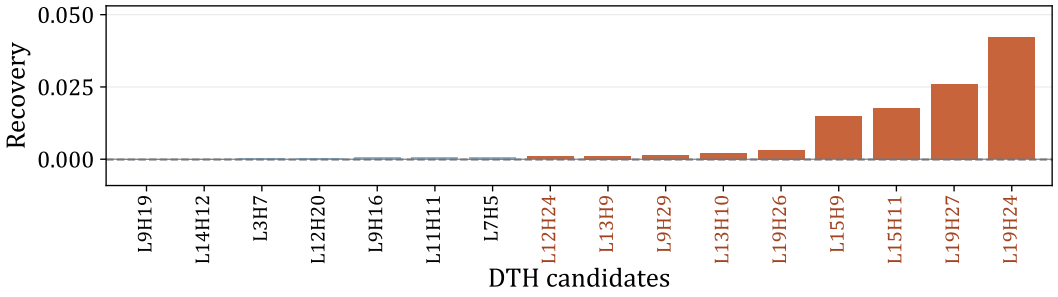

Figure 34: Qwen3-8B, spatial reasoning: 576 pairs (both directions); ranks 43–58/58.

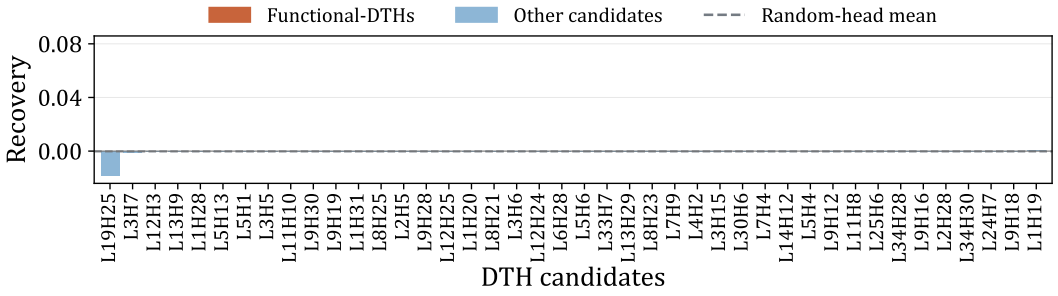

Figure 35: Qwen3-8B, colour-container: 560 pairs (both directions); ranks 1–42/58.

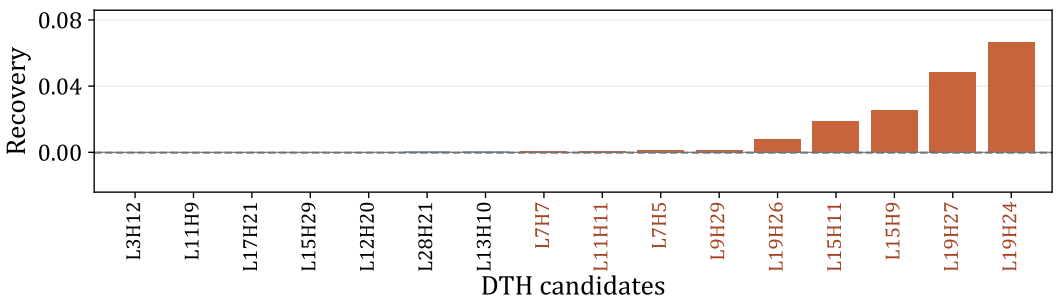

Figure 36: Qwen3-8B, colour-container: 560 pairs (both directions); ranks 43–58/58.

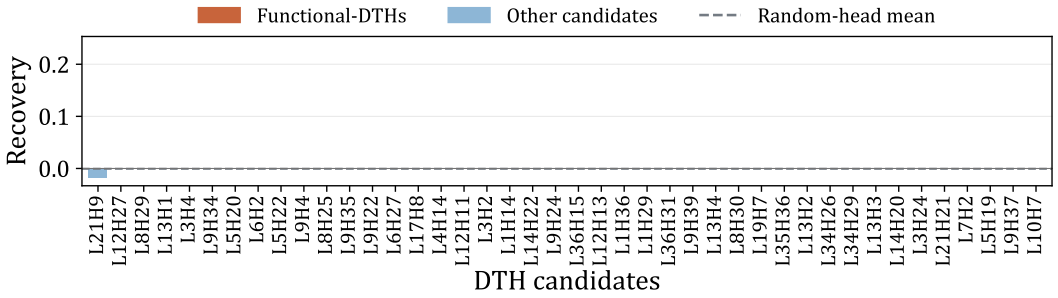

Figure 37: Qwen3-14B, entity tracking: 592 pairs (both directions); ranks 1–42/80.

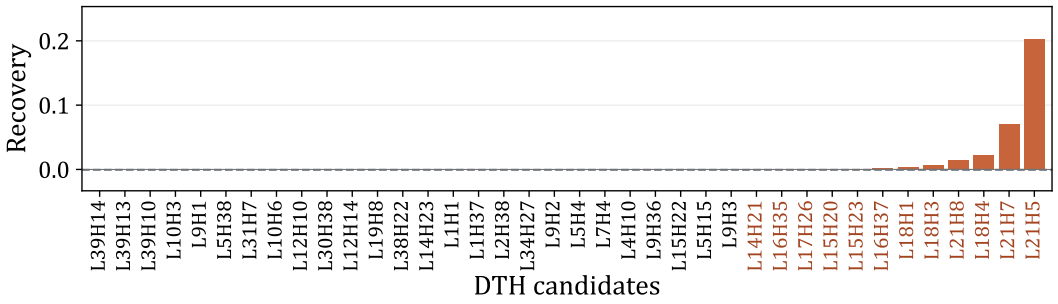

Figure 38: Qwen3-14B, entity tracking: 592 pairs (both directions); ranks 43–80/80.

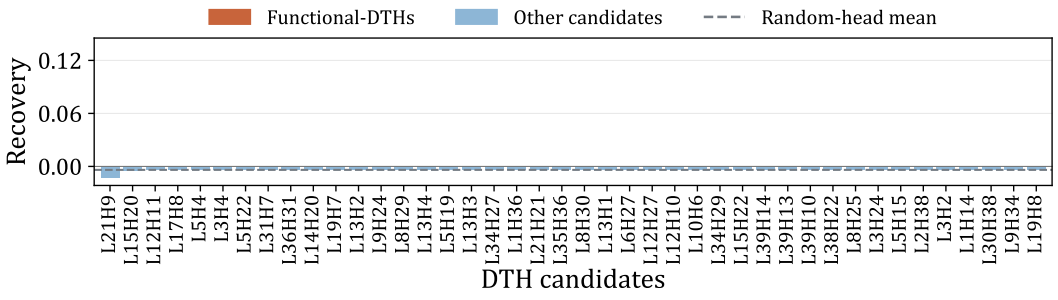

Figure 39: Qwen3-14B, variable binding: 350 pairs (both directions); ranks 1–42/80.

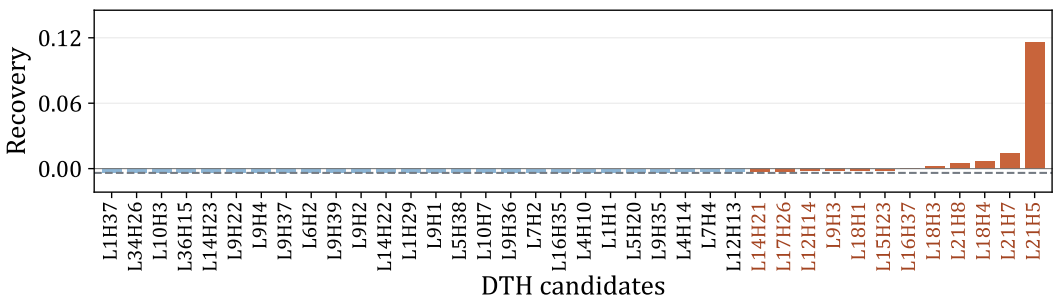

Figure 40: Qwen3-14B, variable binding: 350 pairs (both directions); ranks 43–80/80.

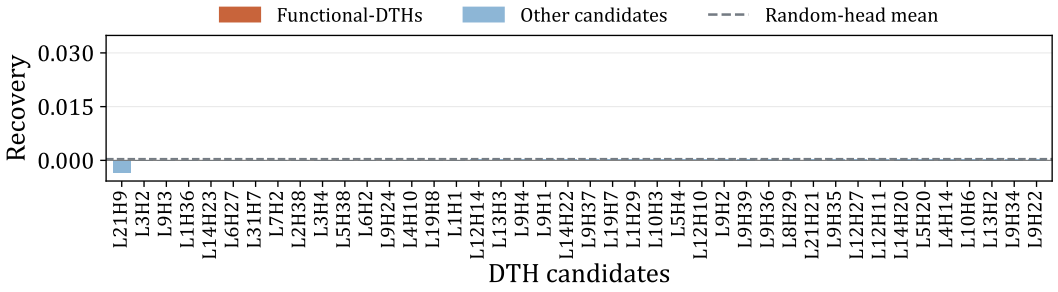

Figure 41: Qwen3-14B, spatial reasoning: 576 pairs (both directions); ranks 1–42/80.

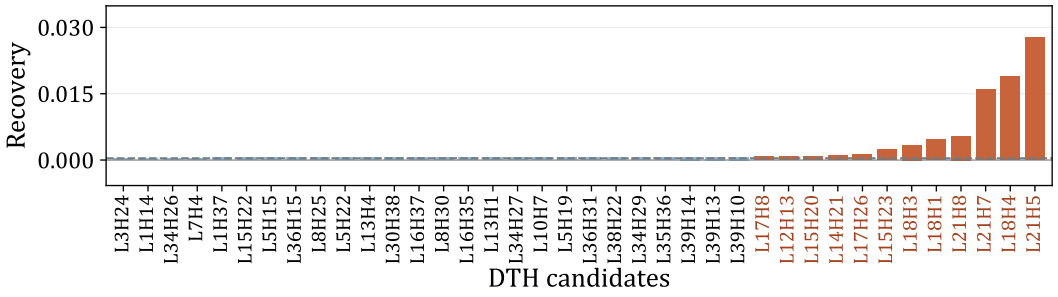

Figure 42: Qwen3-14B, spatial reasoning: 576 pairs (both directions); ranks 43–80/80.

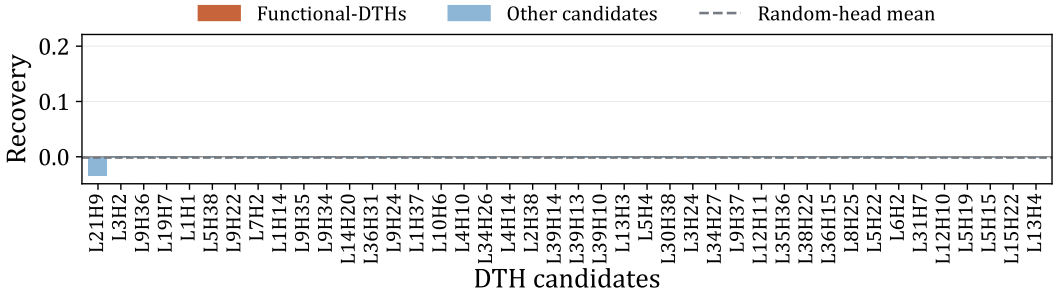

Figure 43: Qwen3-14B, colour-container: 560 pairs (both directions); ranks 1–42/80.

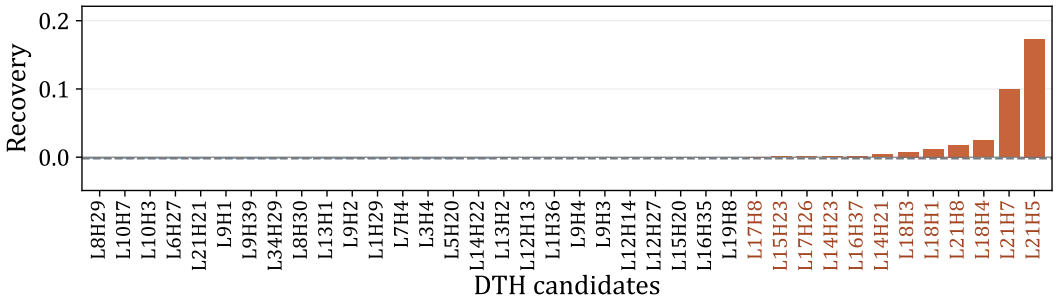

Figure 44: Qwen3-14B, colour-container: 560 pairs (both directions); ranks 43–80/80.

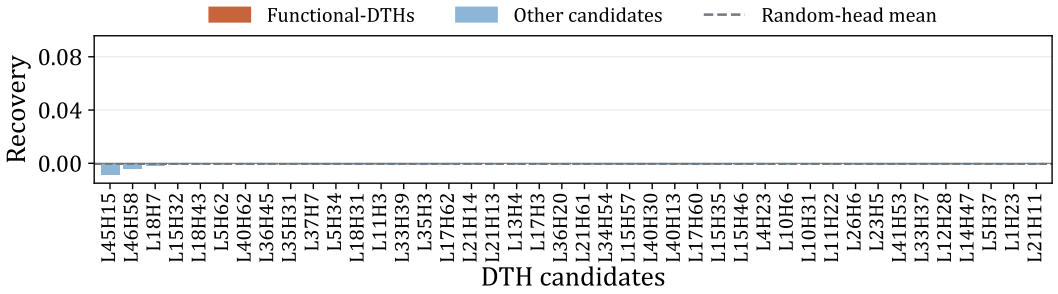

Figure 45: Qwen3-32B, entity tracking: 592 pairs (both directions); ranks 1–42/205.

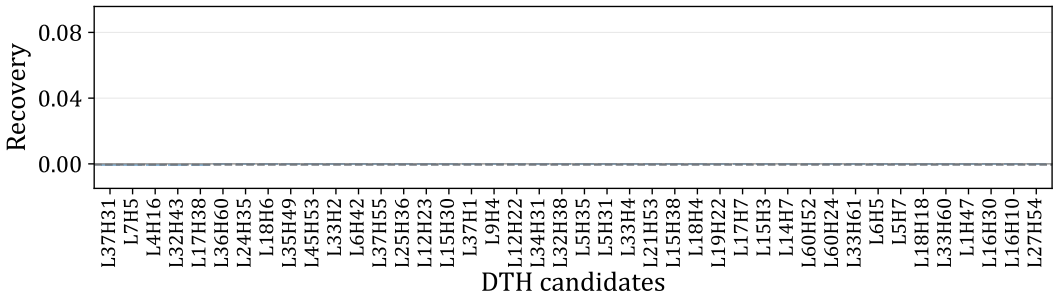

Figure 46: Qwen3-32B, entity tracking: 592 pairs (both directions); ranks 43–84/205.

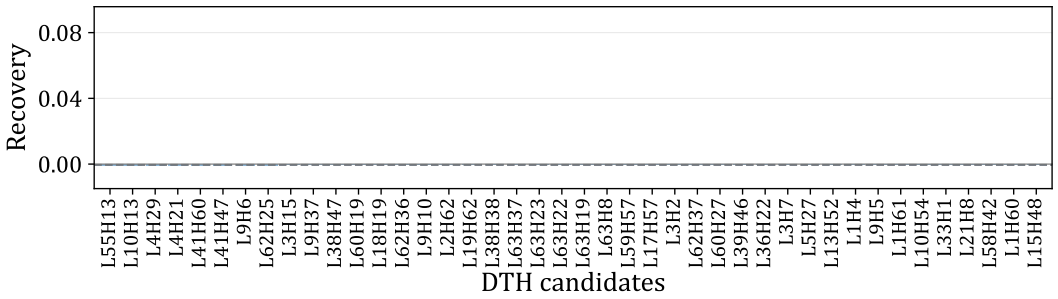

Figure 47: Qwen3-32B, entity tracking: 592 pairs (both directions); ranks 85–126/205.

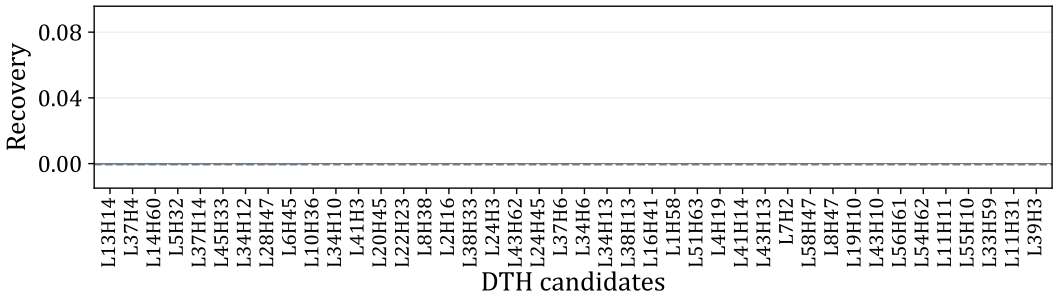

Figure 48: Qwen3-32B, entity tracking: 592 pairs (both directions); ranks 127–168/205.

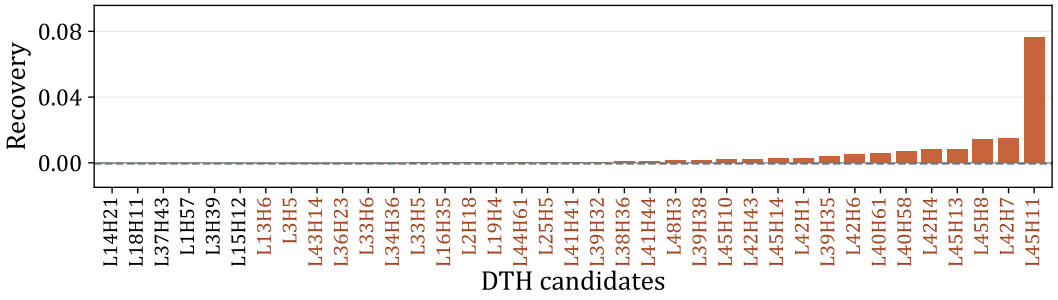

Figure 49: Qwen3-32B, entity tracking: 592 pairs (both directions); ranks 169–205/205.

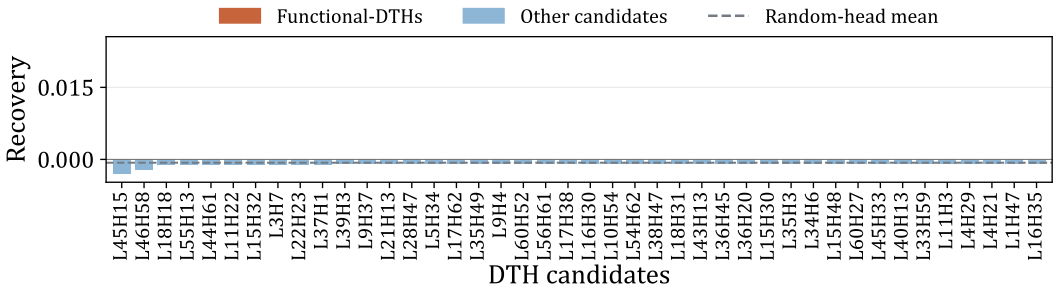

Figure 50: Qwen3-32B, variable binding: 350 pairs (both directions); ranks 1–42/205.

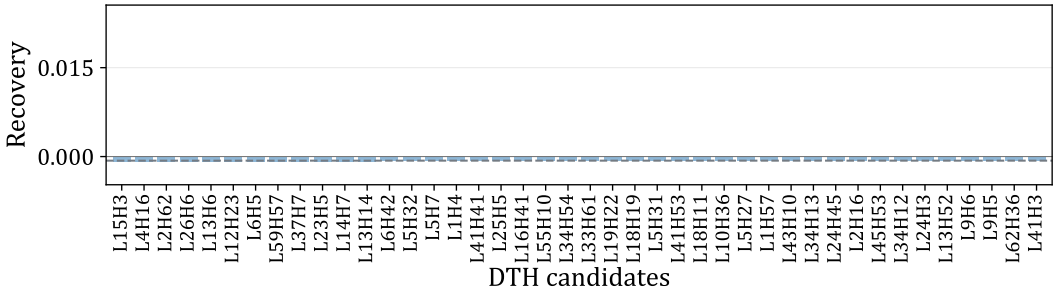

Figure 51: Qwen3-32B, variable binding: 350 pairs (both directions); ranks 43–84/205.

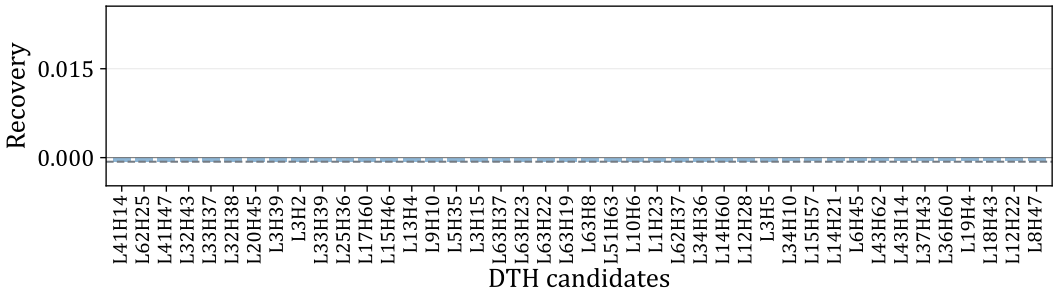

Figure 52: Qwen3-32B, variable binding: 350 pairs (both directions); ranks 85–126/205.

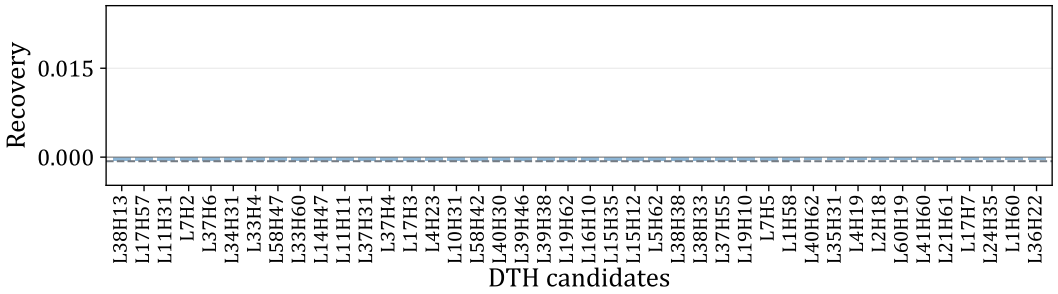

Figure 53: Qwen3-32B, variable binding: 350 pairs (both directions); ranks 127–168/205.

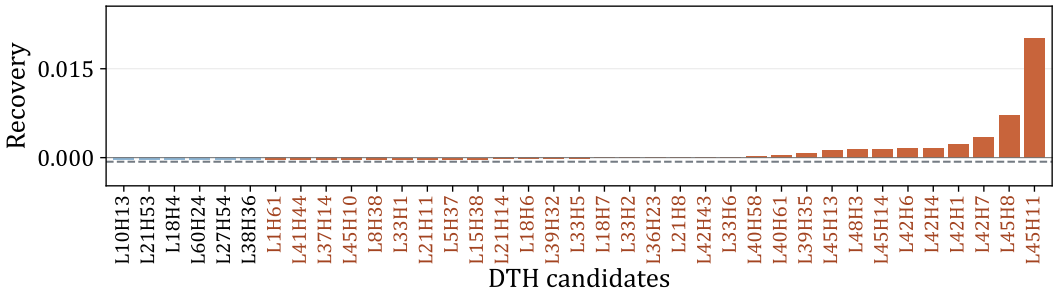

Figure 54: Qwen3-32B, variable binding: 350 pairs (both directions); ranks 169–205/205.

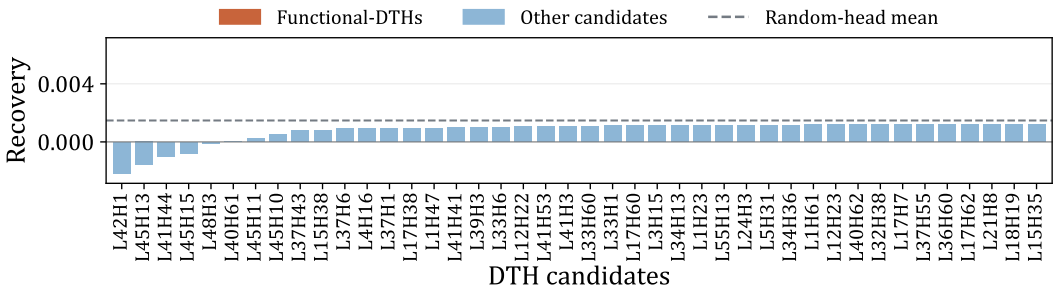

Figure 55: Qwen3-32B, spatial reasoning: 576 pairs (both directions); ranks 1–42/205.

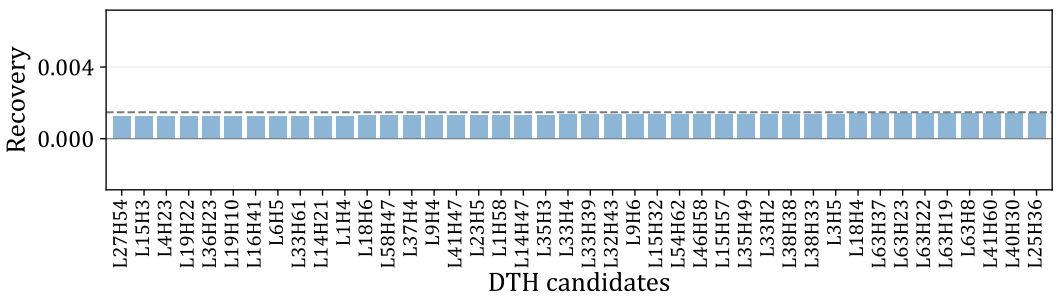

Figure 56: Qwen3-32B, spatial reasoning: 576 pairs (both directions); ranks 43–84/205.

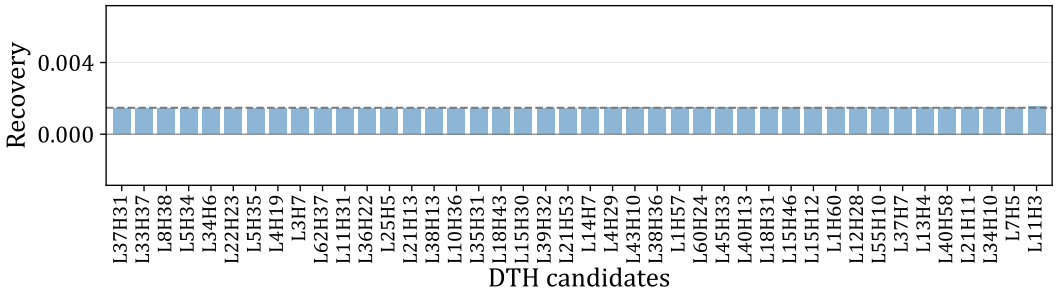

Figure 57: Qwen3-32B, spatial reasoning: 576 pairs (both directions); ranks 85–126/205.

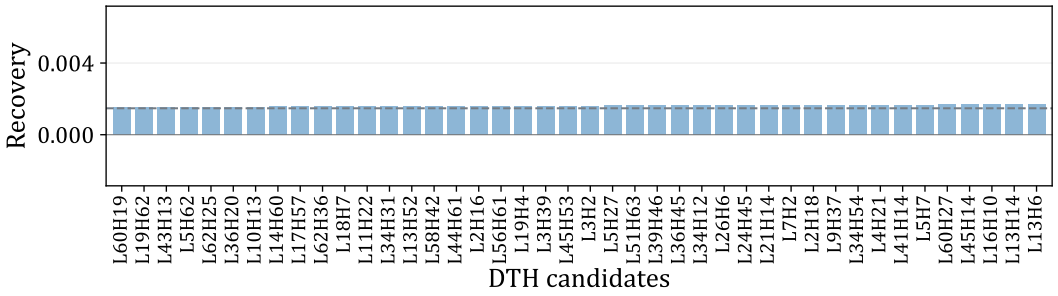

Figure 58: Qwen3-32B, spatial reasoning: 576 pairs (both directions); ranks 127–168/205.

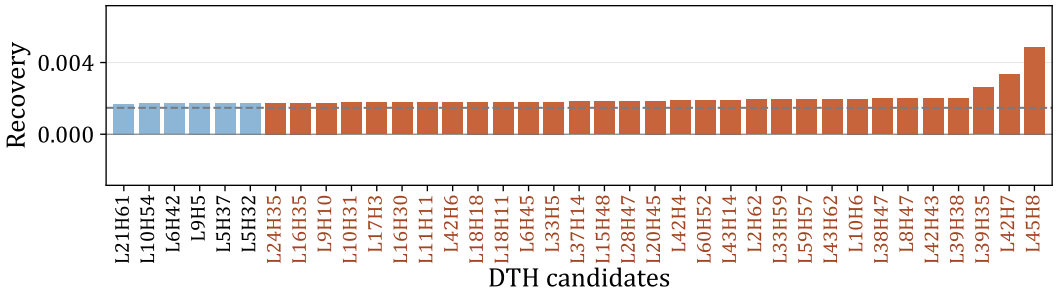

Figure 59: Qwen3-32B, spatial reasoning: 576 pairs (both directions); ranks 169–205/205.

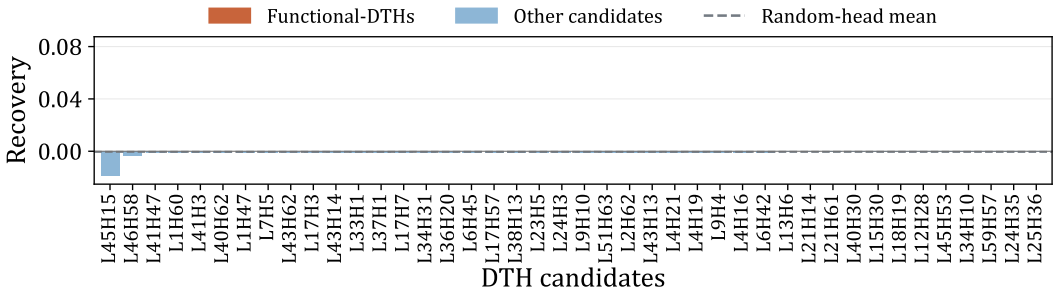

Figure 60: Qwen3-32B, colour-container: 560 pairs (both directions); ranks 1–42/205.

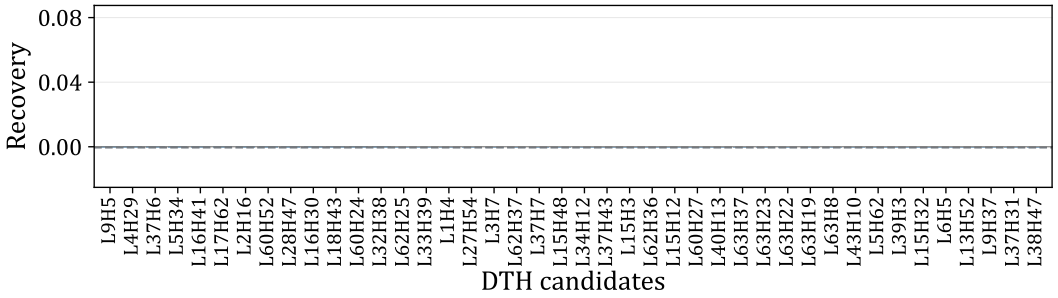

Figure 61: Qwen3-32B, colour-container: 560 pairs (both directions); ranks 43–84/205.

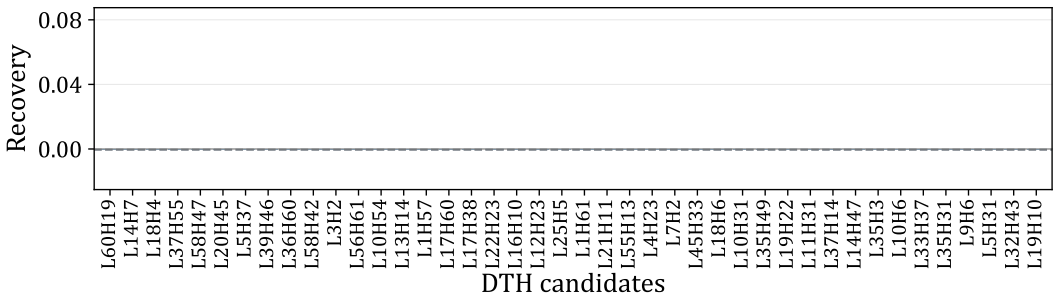

Figure 62: Qwen3-32B, colour-container: 560 pairs (both directions); ranks 85–126/205.

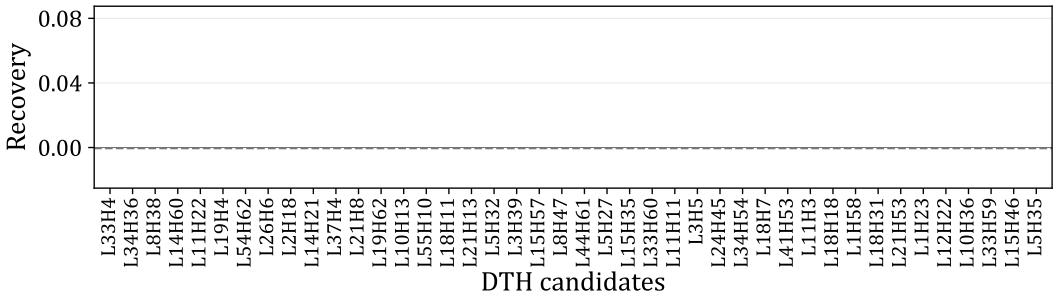

Figure 63: Qwen3-32B, colour-container: 560 pairs (both directions); ranks 127–168/205.

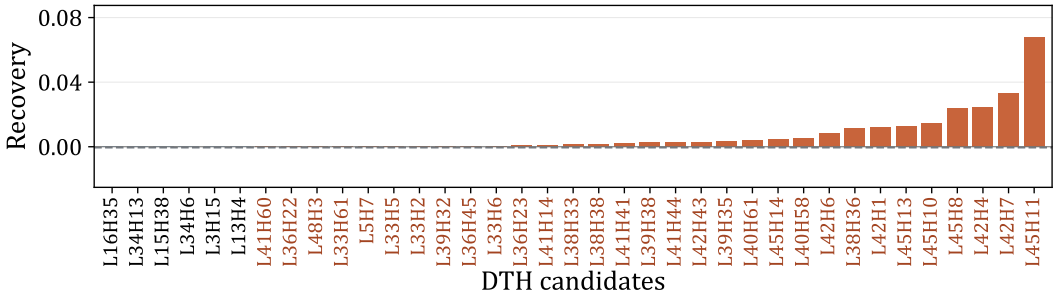

Figure 64: Qwen3-32B, colour-container: 560 pairs (both directions); ranks 169–205/205.

D.3 Cross-Task Overlap of Functional-DTHs

Figure 65 extends Figure 4 to all six models. For independently selected task-specific groups D_A and D_B, shading denotes $|D_A \cap D_B|/|D_A \cup D_B|$; counts show how many heads are shared. This is a descriptive comparison of the frozen selections, not a further selection stage.

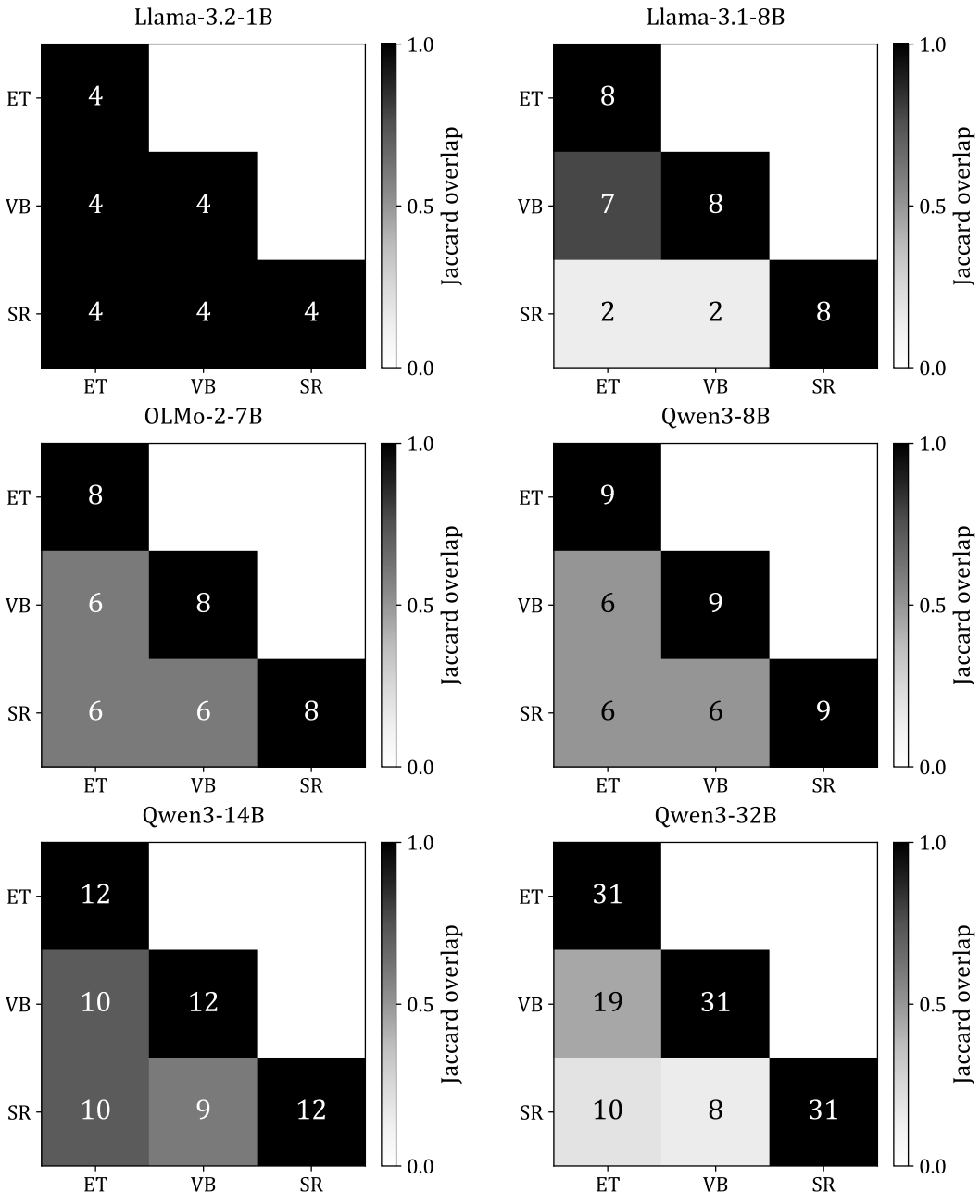

Figure 65: Cross-task overlap of independently selected functional-DTHs. ET, VB, and SR denote entity tracking, variable binding, and spatial reasoning. Shading shows Jaccard overlap; numbers give intersection sizes, with group sizes on the diagonal. Heads are compared only within the same model.

D.4 QK Matching and Channel-Decomposition Results

The following figures extend Figures 5 and 6 to all six models, using functional-DTHs selected independently on the colour-container ranking split. Each model is evaluated on the same 400 QK-split inputs, with seven items and the source anchor in positions 2–6. The split's query colours are `black` and `blue`; nouns are `bag`, `bin`, `box`, and `case`.

For each input, we independently choose between two colours and two nouns at both the source and target, giving 16 complete forward passes. Only these four token positions change; sequence length and positions remain fixed. Four variants belong to each matching condition. We average within a condition first and then equally across inputs for each head, rather than combining query and key vectors from separate inputs.

The matching-effect panels use Equation **??**. The channel-decomposition panels show only Prefix-mismatch, retaining the same full-logit denominator for every channel:

$$D^{01}_{\gamma,\eta} = \frac{\bar{c}^{01}_{\gamma,\eta} - \bar{c}^{11}_{\gamma,\eta}}{\bar{\ell}^{11} - \bar{\ell}^{00}}, \qquad \sum_{\gamma,\eta} D^{01}_{\gamma,\eta} = D^{01}. \tag{9}$$

All heads have a positive mean Both-match minus Neither-match logit gap. Positive bars indicate increases relative to Both-match, and negative bars indicate decreases; black marks give the total logit change. Bars describe per-head effects, not uncertainty intervals; no heads are removed based on the sign or magnitude of their effects. Matching effects and channel changes are shown in consecutive panels. Where a model's heads span multiple panels, each plot type retains the same vertical scale across those panels.

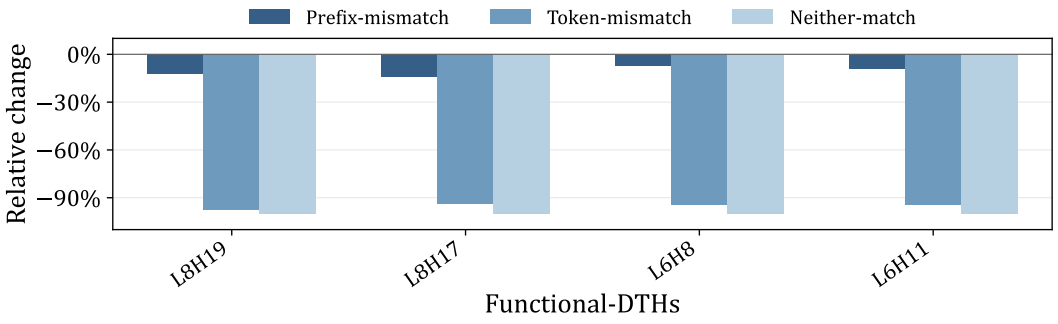

Figure 66: Llama-3.2-1B: matching effects (400 inputs).

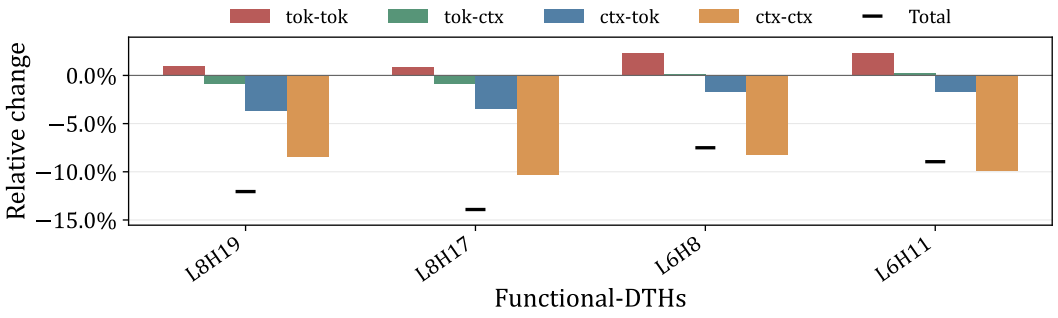

Figure 67: Llama-3.2-1B: Prefix-mismatch channel changes (400 inputs).

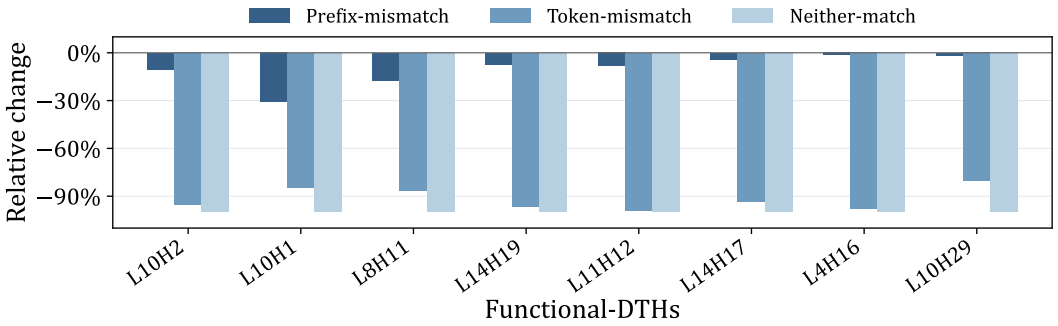

Figure 68: Llama-3.1-8B: matching effects (400 inputs).

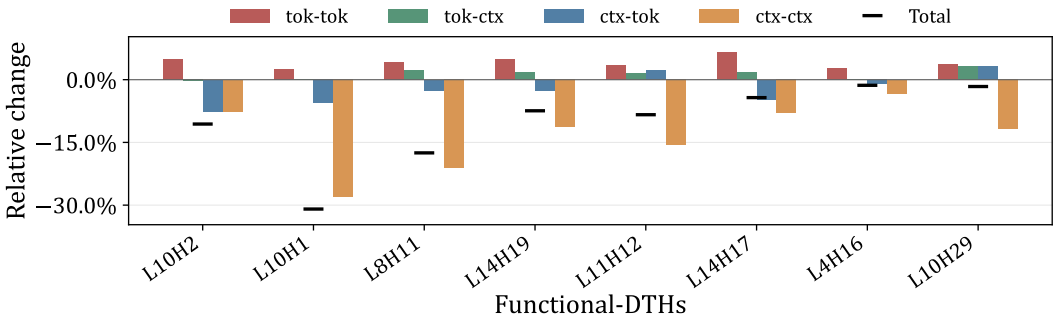

Figure 69: Llama-3.1-8B: Prefix-mismatch channel changes (400 inputs).

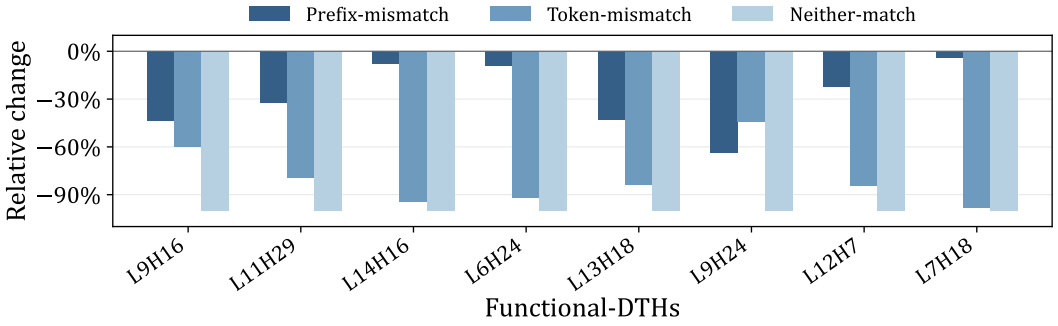

Figure 70: OLMo-2-7B: matching effects (400 inputs).

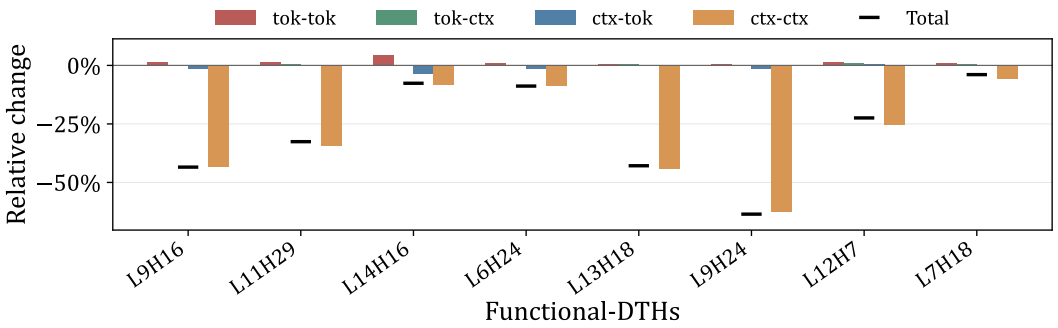

Figure 71: OLMo-2-7B: Prefix-mismatch channel changes (400 inputs).

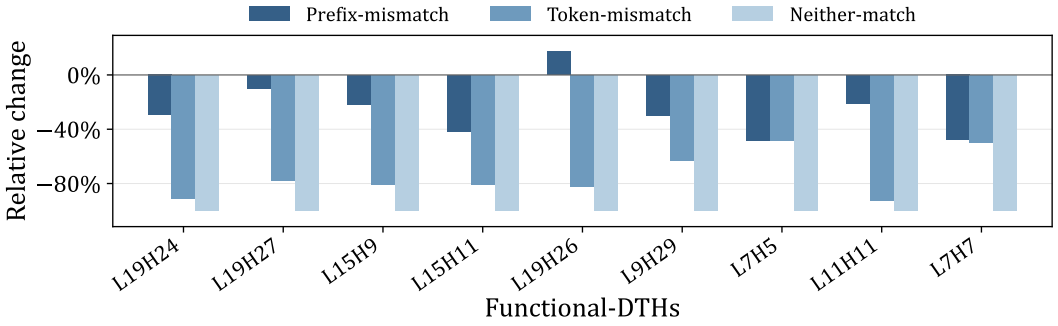

Figure 72: Qwen3-8B: matching effects (400 inputs).

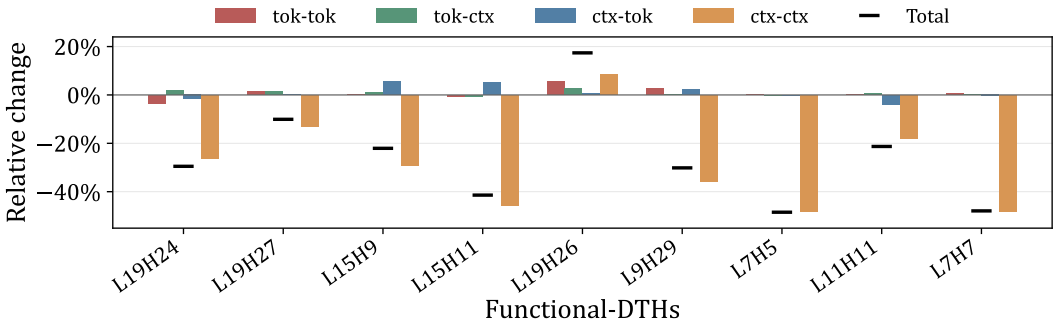

Figure 73: Qwen3-8B: Prefix-mismatch channel changes (400 inputs).

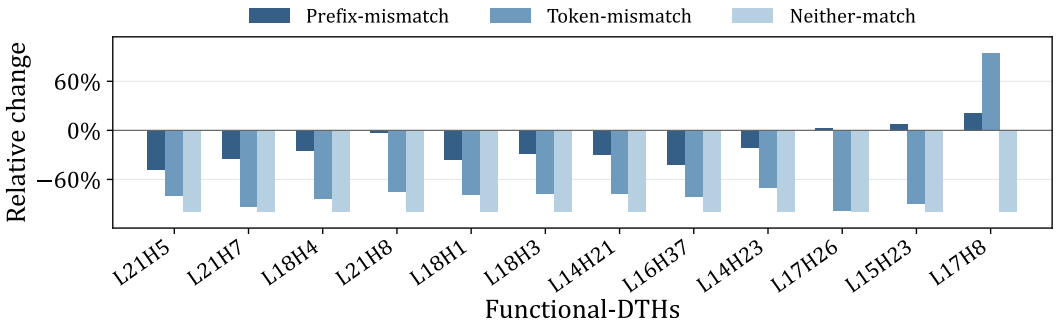

Figure 74: Qwen3-14B: matching effects (400 inputs).

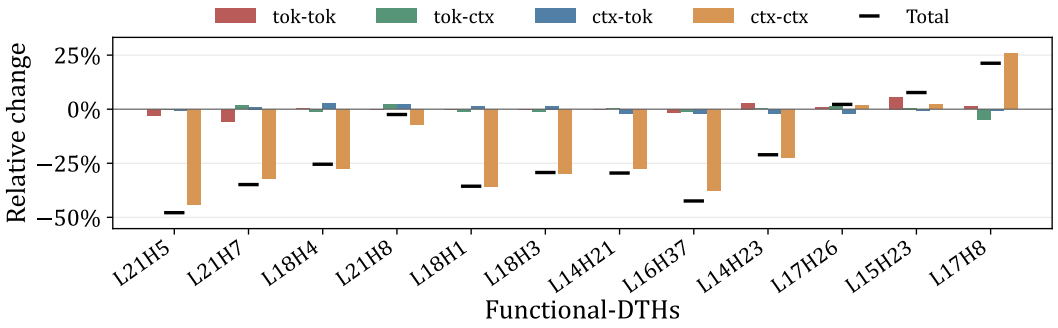

Figure 75: Qwen3-14B: Prefix-mismatch channel changes (400 inputs).

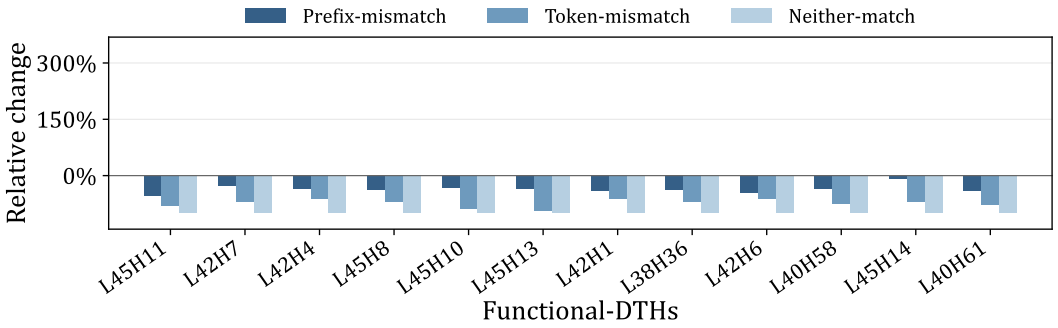

Figure 76: Qwen3-32B: matching effects (400 inputs). Part 1/3.

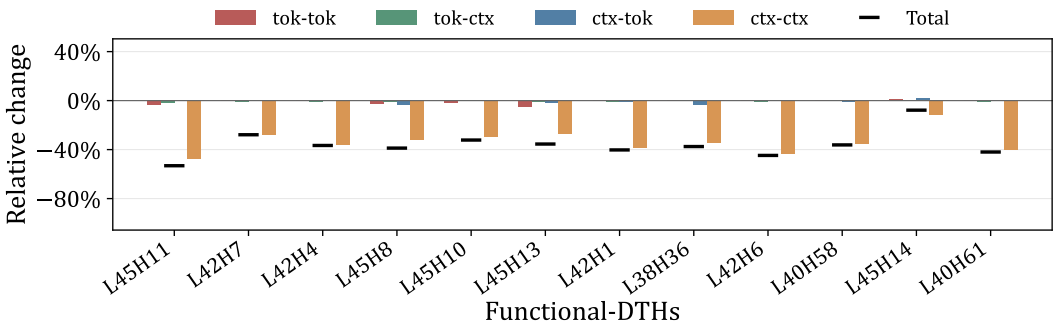

Figure 77: Qwen3-32B: Prefix-mismatch channel changes (400 inputs). Part 1/3.

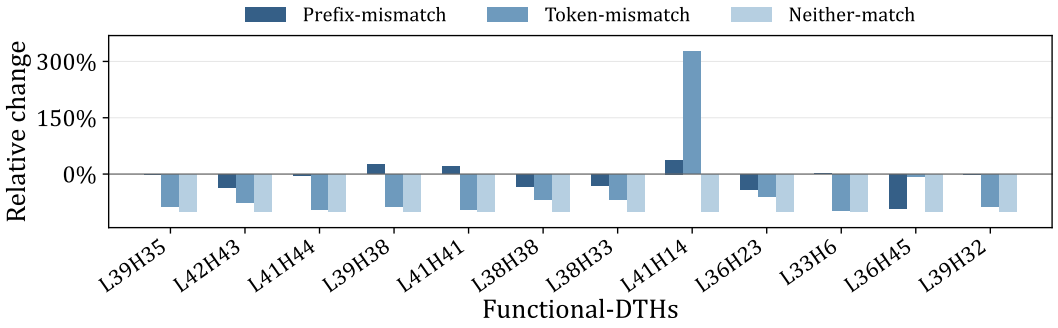

Figure 78: Qwen3-32B: matching effects (400 inputs). Part 2/3.

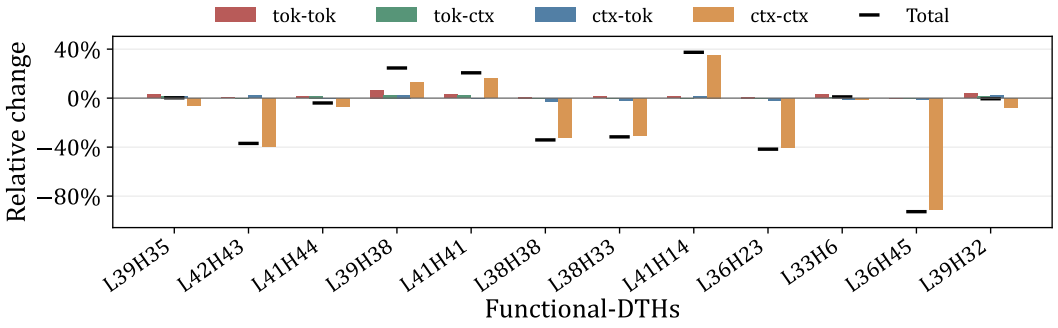

Figure 79: Qwen3-32B: Prefix-mismatch channel changes (400 inputs). Part 2/3.

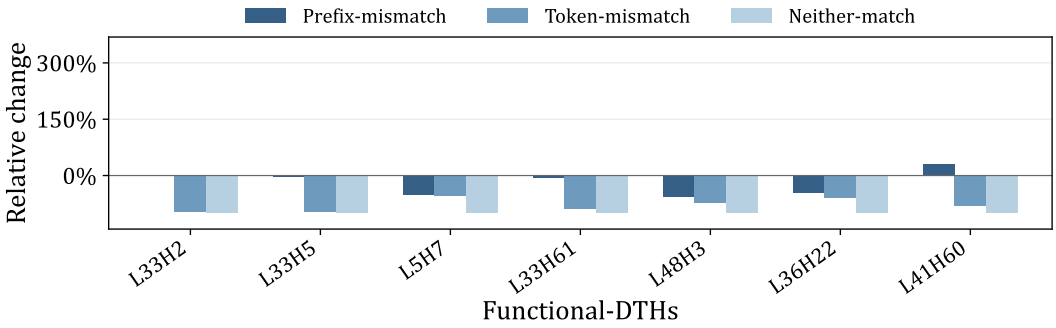

Figure 80: Qwen3-32B: matching effects (400 inputs). Part 3/3.

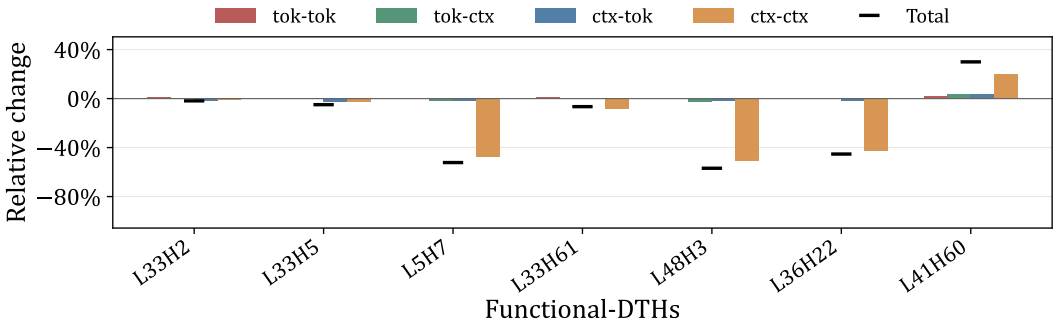

Figure 81: Qwen3-32B: Prefix-mismatch channel changes (400 inputs). Part 3/3.