

Reading Multi-Token Answers from a Single Hidden State

Arnav Raj

Indian Institute of Technology Delhi
arnavvraj.compsci@gmail.com

Abstract

Token lenses such as the logit lens decode a language model’s hidden state into a distribution over the vocabulary, but they read out one token at a time and cannot recover answers that span several tokens, such as “North Korea”. In this paper we ask: given a single hidden state, can the model itself recover the full answer? Our method is a refinement of Patchscopes, which decode a hidden state by patching it into a second prompt. We patch a hidden state from a late layer into an early layer of a copy prompt (a few-shot prompt that asks the model to repeat its input). We find that on Qwen3-14B this recovers the exact answer for 47% of held-out two-word items, up from 21% when the state is patched at the layer it came from. By construction, the state of another item or a random vector does not encode the answer, and patching in either one almost never recovers it. Across two models (Qwen3-14B and Qwen3-8B) and two kinds of answers (two-word phrases and single words or numbers split into sub-word tokens), patching at an early layer roughly doubles exact-match accuracy, and the improvement is also reflected in the results on a new set of test items. On the WorkspaceBench benchmark, a variant of the method recovers 14% of basic-readout answers, versus 0% for token lenses. On typo correction, it outperforms token lenses summarized by an external language model, despite using no training and no such model. Finally, we analyze why patching at an early layer helps: the linear readouts we evaluated fail to decode the second token of an answer, yet causal experiments show that the model’s own layers recover it from the same state.

Code —

<https://github.com/deadmash07/multitoken-readout>

1 Introduction

Consider the prompt “Fact: Pyongyang is the capital of”, which Qwen3-14B completes with “North Korea”. Suppose we retain only the hidden state of the final token at layer 32 of the model’s 40 layers and discard the prompt. Token lenses decode such a state into a ranked list of vocabulary tokens. The logit lens (nostalgebraist 2020) applies the model’s unembedding to the state, and the Jacobian lens (J-lens) is a linear readout released for the Qwen3 models we study (Gurnee et al. 2026). On this example, the logit lens ranks “Korea” second and “North” fifth, and the J-lens ranks both within its top ten. Each lens provides its highest-ranked tokens, as opposed

to an ordered answer string. Under exact-match scoring, and under the regular-expression scorer of the WorkspaceBench interpretability benchmark (Blank et al. 2026b), both lenses score 0% on every multi-token task. The benchmark reports nonzero token-lens scores only after a large language model (LLM) summarizes the top ten tokens into a sentence.

Much of what we would like to read from a hidden state spans several tokens, including names and numbers, chemical compounds, the intermediate (bridge) entity of a two-hop question, and the answer the model is about to give. When a method outputs the exact answer string, we can score it automatically and put its accuracy in context using controls. A ranked list of tokens has to be interpreted by a person or by another language model. Patchscopes (Ghandeharioun et al. 2024) can produce such strings by patching a hidden state into a second prompt, the target prompt, and reading out whatever the model generates. The token-identity Patchscope uses a few-shot *copy prompt* of the form “A -> A; B -> B; ?”, so that the model repeats whatever is placed at the placeholder “?”. In the original work, the state is patched at the same layer it was taken from, its source layer, and only the next token is scored.

We change this setup in one respect, by patching the hidden state into an early layer of the copy prompt, far below its source layer. The model’s remaining layers can then reconstruct the answer before generation begins (Figure 1). We call the layer at which the state is patched the target layer, and we refer to the original and the modified versions as the same-layer and the early-layer Patchscope. On the example above, the same-layer Patchscope produces “North America” and the early-layer Patchscope produces “North Korea”. Both versions are training-free, and each receives only the hidden state and the index of its source layer. Our work makes the following contributions.

- **Exact multi-token readout.** On Qwen3-14B, exact-match accuracy on held-out two-word answers increases from 21% to 47% when we patch at layer 4 instead of at the source layer. The same effect appears for sub-word answers and on Qwen3-8B, and a new test set shows it as well (Section 3).
- **Transfer to WorkspaceBench.** To adapt the method to this benchmark, we use a *continuation prompt*, which patches the state into the last token of a neutral sentence such as “The answer is” and lets the model continue.

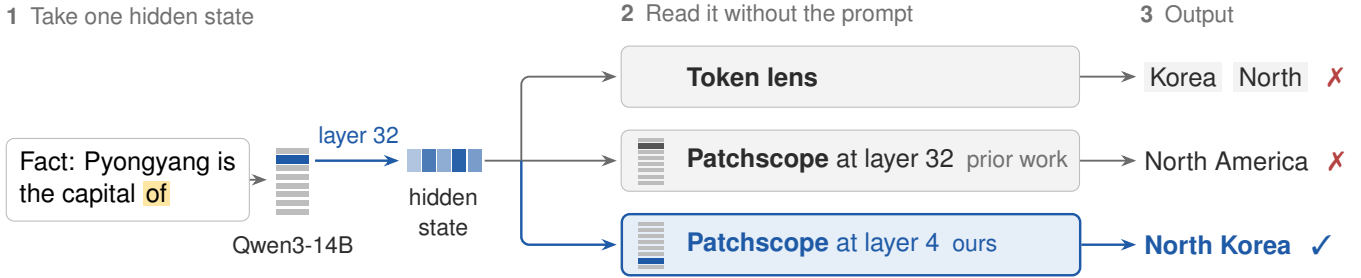


Figure 1: Comparison of three readouts for the hidden state of the final prompt token at layer 32 of Qwen3-14B, which we decode without the prompt. Besides a token lens, we use a Patchscope that patches the state into a copy prompt, either at layer 32 (prior work; Ghandeharioun et al. 2024) or at layer 4 (ours). Only the Patchscope at layer 4 recovers the full answer.

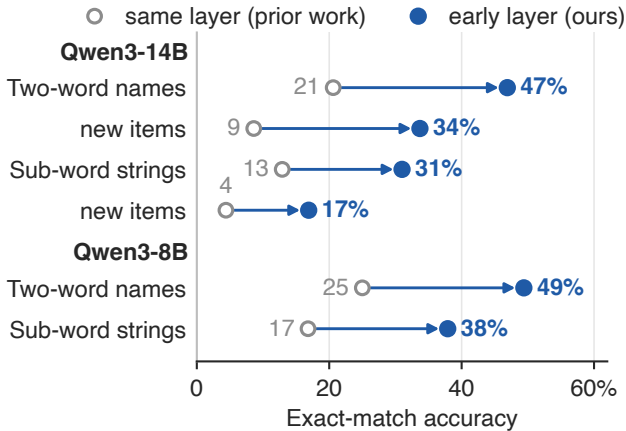


Figure 2: Exact-match accuracy of the same-layer Patchscope (prior work, grey) and the early-layer Patchscope (ours, blue) on held-out items; each arrow shows the improvement. The early-layer Patchscope patches at layer 4 on Qwen3-14B and at layer 8 on Qwen3-8B. *New items* denotes the new test set, whose answers do not overlap with earlier items, benchmark items, or prompt exemplars. An item counts as correct when a majority of the eight prompt variants generate the exact answer.

With the benchmark’s own model and scorer, the method recovers 14% of basic-readout answers, versus zero for token lenses. It also outperforms LLM-summarized token lenses on typo correction (Section 4).

- **Why early patching works.** The model’s own computation recovers the second token of an answer from the hidden state, but the linear readouts we evaluated do not decode it. Section 5 uses this contrast to help explain why patching at an early layer works.

Statement of contributions. This paper presents new, previously unpublished work. The early-layer configuration of the token-identity Patchscope is new, as is its evaluation (Section 3), which uses exact multi-token scoring and item-level controls and includes a new test set and a second model. So are the transfer to an external benchmark (Section 4) and the analysis of why early patching works (Section 5).

2 Method

Hidden states. A transformer maintains a residual stream vector at each token position, which every layer reads from and writes to (Elhage et al. 2021). We refer to the residual stream vector at a given position after a given layer as a hidden state, and we index layers from zero. On Qwen3-14B we read the state after layer 32, at about 80% of its depth.

Token lenses. A token lens maps a hidden state to scores over the vocabulary. The logit lens (nostalgebraist 2020) applies the final normalization and the unembedding to the state. The J-lens (Gurnee et al. 2026) is a linear readout fitted over generic contexts and is released for the Qwen3 models (Yang et al. 2025) we study. Following WorkspaceBench, we take the top ten tokens of each lens as its output.

Patchscopes. A Patchscope (Ghandeharioun et al. 2024) decodes a hidden state through generation. Concretely, we take the hidden state of the final token of the source prompt at source layer ℓ and, in a second run of the same model on a target prompt, replace the residual stream at one position and target layer τ with this state. The model then decodes greedily from the target prompt, and the generated text is the readout. The patch is the intervention used in causal mediation analysis, interchange interventions, and causal tracing (Vig et al. 2020; Geiger et al. 2021; Meng et al. 2022). Zhang and Nanda (2024) and Heimersheim and Nanda (2024) discuss best practices for its use. Prior work uses the token-identity prompt with $\tau = \ell$ (the same-layer Patchscope). We instead set τ far below ℓ (the early-layer Patchscope), with $\ell = 32$ and $\tau = 4$ on Qwen3-14B. Because the target run adds computation of its own, a correct readout shows that the answer can be decoded from the state but does not guarantee that it is stored at the source position (Section 5).

Target prompts. We use two target prompts with eight fixed variants each and treat the eight generations for an item as votes. The copy prompt is the few-shot token-identity prompt of Ghandeharioun et al. (2024) with multi-token exemplars, for example “Grand Rapids -> Grand Rapids; kaleidoscope -> kaleidoscope; 3817 -> 3817; ?”. The state replaces the residual stream at the placeholder token “?”, and the model then generates eight tokens. We truncate the output at the first “;”, “->”, or newline. The continuation prompt is a neutral prefix in which the state replaces the residual stream

at the final token, and the model then generates twelve tokens. Four of its variants are prose prefixes that end mid-clause, such as “She opened the letter slowly and read the first line, which mentioned the”. The other four, which we call answer frames, are prefixes such as “The answer is”. The target prompts do not contain any answer string (Appendix A lists all variants).

Data. Each item is a cloze-style prompt with a multi-token answer, of the form “Fact: Pyongyang is the capital of”. Two-word items have answers that name people, places, compounds, or events, for example “Mexico City”. Sub-word items have answers that the tokenizer splits into several pieces, such as rare words, numbers, and translations. We split each item set into a tuning half and a held-out half. All results are on held-out items that the model answers correctly under greedy decoding (175 two-word and 342 sub-word items for Qwen3-14B). We constructed a new test set of 1,002 items whose answers do not overlap with earlier items, benchmark items, or prompt exemplars (Appendix B). Section 5 also uses two-hop items such as “Fact: The continent of the country whose capital is Riyadh is”, in which the intermediate (bridge) entity is “Saudi Arabia”. We call the final token of the entity named in the prompt (“Riyadh”) the first-hop token.

Metrics and controls. Before scoring, we normalize case and whitespace in each generation and in the answer. An item counts as correct when a majority (more than four) of the eight prompt variants generate the exact answer, and we report the fraction of correct items as exact-match accuracy. Token lenses score 0% under this metric by construction, since they return single tokens. We also evaluate each Patchscope with three substitute states (Appendix C): the state of another item from the same category (the shuffled control), the state of an item from a different category (the cross-category control), and a Gaussian vector with the norm of the original state (the random control). Analogous to control tasks for probes (Hewitt and Liang 2019), these controls measure the accuracy that the target prompt produces without item-specific information. On WorkspaceBench, we use the benchmark’s items, model, and scorer without modification.

3 Exact Multi-Token Readout

We first evaluate whether the early-layer Patchscope decodes multi-token answers from a single hidden state. We apply the copy prompt to the 175 held-out two-word items of Qwen3-14B, with source layer 32 and target layer either 32 (the same-layer Patchscope of Ghandeharioun et al. 2024) or 4 (the early-layer Patchscope). Results are shown in Figure 2.

The same-layer Patchscope achieves 21% exact-match accuracy, and patching at layer 4 increases accuracy to 47%. For every model and item set, accuracy under the three controls stays at or below 0.7% (Appendix D). The effect varies from item to item, as Table 1 shows for eight held-out items drawn at random. Several of these items are missed by every prompt variant with the same-layer Patchscope and recovered by all variants with the early-layer Patchscope, while others are missed by both Patchscopes.

Answer	Same layer	Early layer (ours)
Edward Jenner	Edward ×	Edward Jenner ✓
Isaac Newton	Isaac ×	Isaac ×
one hundred	100 ×	100 ×
Marilyn Monroe	born in 1980 ×	born ×
United Kingdom	United Kingdom ✓	United Kingdom ✓
Nova Scotia	Nova Scotia ✓	Nova Scotia ✓
Johannes Kepler	Johannes ×	Kepler ×
Mother Teresa	Mother ×	Mother Teresa ✓

Table 1: Example outputs for eight randomly sampled held-out items (Qwen3-14B, source layer 32, copy prompt). For each item and target layer, we show the output that the eight prompt variants generated most often. ✓ marks the correct answer and × an incorrect output.

To test for overfitting to the items used during development, we evaluate on the new test set, whose answers do not overlap with those items, benchmark items, or prompt exemplars. On this set, the early-layer Patchscope again outperforms the same-layer Patchscope on both two-word and sub-word answers (Figure 2). The improvement on two-word answers is 25 percentage points, which matches the 26 points on the original items.

On Qwen3-8B, a target layer of 4 fails because the copy prompt reproduces its own exemplars, and with target layer 8 the early-layer Patchscope again roughly doubles accuracy (Figure 2). The effect appears on a second model as well, but at a different target layer. On Qwen3-1.7B, the improvement is smaller (Appendix D).

We trace the improvement to answer tokens beyond the first. For single-token answers, the same-layer Patchscope already recovers most answers, and the early-layer Patchscope performs slightly worse. We split the two-word items by whether the first answer token alone places the second in the model’s top ten. On the hard items, where it does not, accuracy rises from 2% with the same-layer Patchscope to 34% with the early-layer Patchscope (Appendix D). Both results may be explained by the same-layer Patchscope copying the first token and relying on the model to complete it, and by the early-layer Patchscope reconstructing the full answer.

4 Evaluation on WorkspaceBench

We next evaluate the early-layer Patchscope on WorkspaceBench (Blank et al. 2026b), an external benchmark with its own items, model, and scorer. The benchmark provides the hidden state of a prompt and counts a readout as a pass when it contains the answer under a fixed regular expression. We report the fraction of passing items as the pass rate. We use three task families: basic readout (a factual question), typo correction (the correct spelling of a misspelled word), and multihop (a two-hop question, scored on its final answer and its bridge entity). The benchmark model is Qwen3.6-27B, and the benchmark repository reports results for raw token lenses, LLM-summarized token lenses, and two readouts trained on that model (a natural-language autoencoder and an oracle lens).

The benchmark reads the final prompt token, which is

Method	Requires	Basic	Typo	Multihop
J-lens	–	0	0	0
Logit lens	–	0	0	0
J-lens + LLM summary [†]	ext. LLM	24	17	8
Logit lens + LLM summary [†]	ext. LLM	20	20	8
Ours	–	14	21	2
Ours, best layer*	–	22	33	6
Autoencoder [†]	training	53	75	22
Oracle lens [†]	training	71	83	32

Table 2: WorkspaceBench pass rate (% , higher is better) on Qwen3.6-27B, with 100 items per task. *Requires* lists what each method needs in addition to the hidden state: an external language model (ext. LLM), which rewrites the lens’s top ten tokens into a sentence before scoring, or a decoder trained on the benchmark state model (training). A dash marks methods that use the hidden state alone. *Ours* is the continuation prompt with source layer 52 and target layer 8. Bold marks the best training-free method on each task. [†]Results reported by Blank et al. (2026b). *Best of three target layers.

usually a function word such as “the” or “is”. At this position the readout must predict the upcoming answer, and the copy prompt performs poorly. We instead use the continuation prompt, in which the model continues generating after the patched position. In a pilot on Qwen3-14B, raw token lenses fail to recover any basic-readout answer, as do the prompt without a patch and the shuffled, cross-category, and random controls (Appendix D).

Table 2 reports results on Qwen3.6-27B. Our default configuration (source layer 52, target layer 8) recovers 14% of basic-readout answers, versus 0% for raw token lenses. Without an external LLM, the Patchscope outperforms the published LLM-summarized token lenses on typo correction and approaches them on basic readout. The summarized lenses use such an LLM to rewrite their top ten tokens, while our method is scored on its eight short generations directly. The two readouts trained on the benchmark model score higher on basic readout and typo correction. Multihop questions remain difficult for all training-free methods.

Most correct readouts come from the answer frames, and most correct typo-correction readouts come from a single frame, “The correct term for it is” (Appendix D). The answers do not originate in the prompts. Without a patch, the prompts fail to produce a correct answer, and when the state of a different item is patched in, the model generates that item’s answer. A correct generation names the answer in running text, as in “methyl salicylate was approved for use in the United States” for a basic-readout item on Qwen3-14B.

5 Analysis

Why does an early target layer help, and what does a correct readout imply? We study the two questions with linear readouts and patching.

Linear readouts. We first tried to extend the J-lens to token sequences with linear maps. The linear readouts we evaluated

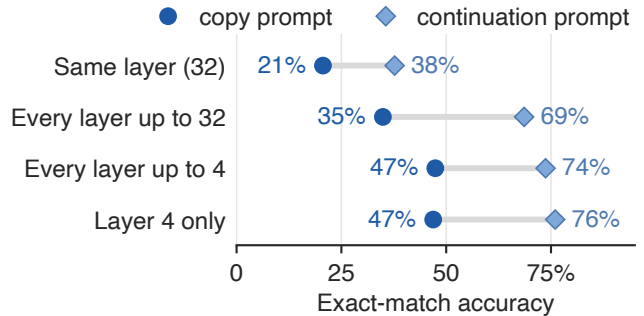


Figure 3: Effect of where the state is patched. Exact-match accuracy of the copy and continuation prompts on Qwen3-14B (source layer 32, 175 two-word items) with the state patched at the source layer only, every layer up to 32, every layer up to 4, or layer 4 only. Patching every layer up to 32 recovers part of the improvement of the early-layer Patchscope, and patching every layer up to 4 matches the single patch at layer 4.

do not decode the second answer token. The step readout extends the J-lens by one position. Concretely, given the state and the true first answer token, it predicts the second token with a first-order (linear) approximation of the model. It ranks the correct second token first for at most one item per layer. We also fit a linear probe in the style of Future Lens (Pal et al. 2023), a ridge regression trained on generic text to map a hidden state to the hidden state one position ahead. On the second token of sub-word answers, the probe performs at the level of its shuffled control (Appendix D). Because the same probe accurately predicts the first token of single-token answers, the failure is specific to the second token. Decomposing the released J-lens into a same-position term, which predicts the current position, and future terms, which predict later positions, suggests a reason for this failure. With depth, the same-position term dominates, and the future terms place most of their weight on positions far beyond the next token (Appendix D).

The second token is nevertheless decodable from the state through the model’s own computation. In a one-position patch, we patch the state into a generic prompt at the position before the first answer token, append that token, and let the model predict the next one. This patch places the correct second token in its top ten for 44% of sub-word items, versus 13% with the state of another item. Its first-order (tangent) approximation reaches only 7%. This gap suggests that the second token is encoded in a form that the model’s own layers can read and a linear map cannot. It is also consistent with our explanation of the early-layer Patchscope, in which the subsequent layers can reconstruct the token before generation begins.

Patching lower layers. We posit two hypotheses for the poor performance of the same-layer Patchscope. The first, proposed by Ghandeharioun et al. (2024, App. C), is that the representation of the placeholder token persists in the layers below the patch and the generation attends to it. The second

is that the layers between the target layer and the source layer perform computation that the answer requires. We test these hypotheses by patching the state into every layer up to the source layer, which removes the placeholder representation from all lower layers. If the first hypothesis is correct, this should recover the full improvement of the early-layer Patchscope, and if the second is correct, accuracy should stay at the level of the same-layer Patchscope. Figure 3 shows an intermediate outcome, in which patching every layer up to 32 recovers about half of the improvement for the copy prompt and most of it for the continuation prompt. Patching every layer up to 4 matches the single patch at layer 4. The result shows that the lower-layer state at the patch position matters, though it does not distinguish the two hypotheses. One further observation bears on the first hypothesis, namely that the continuation prompt, whose final token is an ordinary word instead of a placeholder, still benefits from an early target layer. Thus, the placeholder token alone cannot explain the poor performance of the same-layer Patchscope.

Bridge entities. While exact recovery shows that the state contains enough information about the item for the Patchscope to generate the answer, it remains unclear whether the model computed the answer at the source position. To examine this distinction, we use bridge entities, which prior work locates at the first-hop token of a two-hop question (Yang et al. 2024; Biran et al. 2024). With the state read at that token (“Riyadh”), the continuation prompt generates “is the capital of Saudi Arabia” for 87% of items. In a neutral-context control, where the same entities appear in sentences that do not ask about their country (“Fact: Yesterday someone mentioned Riyadh is”), the continuation prompt still names the country for 72% of items. This control accounts for most of the effect. The paired difference is small and depends on the template (Appendix D), which is consistent with the Patchscope recomputing the country from the entity using the model’s own knowledge. On the multihop items of WorkspaceBench, the Patchscope decodes the final answer at the last token but almost never decodes the bridge entity at any scanned position (Appendix D).

6 Related Work

Token lenses. The logit lens (nostalgebraist 2020) and the tuned lens (Belrose et al. 2023) map hidden states to the vocabulary with a fixed or learned transformation, and the J-lens (Gurnee et al. 2026) is a linear readout averaged over generic contexts. These lenses are inexpensive and well suited to identifying the single token that a state favors. Their output is a ranked list of tokens instead of a string, so they score 0% on the multi-token tasks of WorkspaceBench (Blank et al. 2026b) unless an LLM summarizes it. Our decomposition of the J-lens into same-position and future terms (Appendix D) suggests why linear extensions to token sequences do not help, and Yan et al. (2026) give a mathematical account of the same split. The R-lens (Blank et al. 2026a) makes single-token readouts more faithful at early layers, which complements our focus on multi-token strings from late layers. Vocabulary projections have also been used to interpret feed-forward updates and weights (Geva et al. 2022; Dar et al.

2023), and a learned linear map can approximate the computation of intermediate layers (Yom Din et al. 2024). Probes (Alain and Bengio 2016; Belinkov 2022) decode a state into a label and sparse autoencoders (Bricken et al. 2023; Huben et al. 2024; Templeton et al. 2024) decompose it into features, while our method decodes it into the answer string.

Decoding through generation. Patchscopes (Ghandeharion et al. 2024) introduced the approach we build on, in which a source state is patched into a chosen prompt, position, and layer, and the model’s generation is read. Their token-identity prompt uses a target layer equal to the source layer and is scored on the next token. Their analysis attributes the drop at late target layers to contamination by the placeholder token, and it notes that the computation of the target model can contribute to the readout. We adopt the second point in treating a correct readout as evidence of decodability (Section 2). SelfIE (Chen et al. 2024) patches the state into a fixed early layer of an instruction prompt and measures token-level treatment effects to separate state-driven text from autoregressive completion. Future Lens (Pal et al. 2023) patches at the source layer into fixed or learned prompts and scores each future token separately, and the linear probe of Section 5 is a readout of the same kind. Earlier target layers have been found to help in other settings, including attribute extraction with Patchscopes, SelfIE, and back-patching (Biran et al. 2024). Unlike these studies, we evaluate the token-identity prompt with target layers far below the source layer under exact multi-token scoring with item-level controls, a combination we have not found in earlier work. Morand et al. (2025) train a decoder that injects one hidden state at the embedding layer and generates the multi-token name of the entity it encodes. Our method needs no training and decodes the answer instead of the input entity. Readers that generate text can also answer from their own knowledge instead of from the patched state (Li et al. 2026; Gong et al. 2026b), which is why we treat exact recovery as decodability and report item-level controls. Trained readouts, including InversionView (Huang et al. 2024), LatentQA (Pan et al. 2026), and the oracle lens of the benchmark, learn a decoder, and the trained readouts of the benchmark score higher than our method (Table 2). Gong et al. (2026a) recover multi-token concepts by using token-level lens outputs to guide a search over candidate concepts, which the model then scores relative to the original activation. In contrast, our method is training-free and uses only the hidden state.

Multi-token representations. Language models assemble the pieces of a multi-token word into a word-level representation at its final piece (Feucht et al. 2024; Kaplan et al. 2025). Hidden states also encode information about tokens beyond the next (Pal et al. 2023), either as preparation for later use or as a by-product of next-token prediction (Wu et al. 2024). Attribution graphs of a production model show an intermediate entity during two-hop completion and a rhyming word chosen before the line that ends with it (Lindsey et al. 2025). We study the reverse direction by recovering the exact answer string from the information in a single state.

Latent multi-hop reasoning. Geva et al. (2023) describe factual recall in two stages, in which subject information is aggregated at the last subject token and attention then propagates the attribute to the final position, the position that WorkspaceBench reads. Yang et al. (2024) and Biran et al. (2024) locate the bridge entity of two-hop questions at the first-hop token. Closest to our pairing of a late source layer with an early target layer, Biran et al. (2024) also show that patching a later state into an earlier layer of the same prompt corrects errors. Liu et al. (2026) also read bridge entities with Patchscopes, without a neutral-context control. Our bridge-entity results are consistent with these findings, because they concern only what our Patchscope decodes beyond the entity itself and leave open what the model represents at the first-hop token.

7 Limitations and Future Work

Our experiments use models from the Qwen3 family, including the benchmark model, and our findings may not generalize to other model families. The effective target layer differs between models (layer 4 for Qwen3-14B and layer 8 for Qwen3-8B), and a rule for choosing it without a small search remains to be found. Finally, while exact recovery shows that an answer is decodable from a hidden state, counterfactual or nonce facts would test whether the model also stores the answer at the source position.

8 Conclusion

Our results show that patching the hidden state into an early layer of the target prompt, instead of its source layer, lets the token-identity Patchscope decode exact multi-token answers that token lenses and the linear readouts we evaluated cannot. The improvement holds on held-out items and on a new test set, and we observe it on a second model as well. With the continuation prompt, the method transfers to an external benchmark evaluated on its own model and scorer. The readout shows that a single late hidden state contains enough item-specific information for the subsequent layers of the model to reconstruct the answer. We hope that the early-layer configuration and the exact-match protocol with item-level controls will be useful for decoding multi-token content from hidden states.

References

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes. *arXiv preprint arXiv:1610.01644*, 2016. doi: 10.48550/arXiv.1610.01644. URL <https://arxiv.org/abs/1610.01644>.

Yonatan Belinkov. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1): 207–219, 2022. doi: 10.1162/coli_a_00422. URL https://doi.org/10.1162/coli_a_00422.

Nora Belrose, Igor Ostrovsky, Lev McKinney, Zach Furman, Logan Smith, Danny Halawi, Stella Biderman, and Jacob Steinhardt. Eliciting latent predictions from transformers with the tuned lens. *arXiv preprint arXiv:2303.08112*, 2023. doi: 10.48550/arXiv.2303.08112. URL <https://arxiv.org/abs/2303.08112>.

Eden Biran, Daniela Gottesman, Sohee Yang, Mor Geva, and Amir Globerson. Hopping too late: Exploring the limitations of large language models on multi-hop queries. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 14113–14130, 2024. doi: 10.18653/v1/2024.emnlp-main.781. URL <https://aclanthology.org/2024.emnlp-main.781/>. arXiv:2406.12775.

Camila Blank, Agam Bhatia, and Neel Nanda. R-lens: Making J-lens more faithful on early layers. *LessWrong*, 5 August 2026, 2026a. URL <https://www.lesswrong.com/posts/nv8oedrnlXKRzNEL9/r-lens-making-j-lens-more-faithful-on-early-layers>.

Camila Blank, Agam Bhatia, Euan Ong, and Neel Nanda. WorkspaceBench: Evaluating interpretability methods for the global workspace. *LessWrong*, 23 September 2026, 2026b. URL <https://www.lesswrong.com/posts/Zeg2JztbdhguL48uH/workspacebench-evaluating-interpretability-methods-for-the>. Code and item banks at <https://github.com/camilablank/workspace-bench> (commit 92d763e).

Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermy, Tom Conerly, Nicholas L. Turner, Cem Anil, Carson Denison, Amanda Askell, Robert Lasenby, Yifan Wu, Shauna Kravec, Nicholas Schiefer, Tim Maxwell, Nicholas Joseph, Alex Tamkin, Karina Nguyen, Brayden McLean, Josiah E. Burke, Tristan Hume, Shan Carter, Tom Henighan, and Chris Olah. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023. URL <https://transformer-circuits.pub/2023/monosemantic-features/index.html>.

Haozhe Chen, Carl Vondrick, and Chengzhi Mao. SelfIE: Self-interpretation of large language model embeddings. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pages 7373–7388. PMLR, 2024. URL <https://proceedings.mlr.press/v235/chen24ao.html>. arXiv:2403.10949.

Guy Dar, Mor Geva, Ankit Gupta, and Jonathan Berant. Analyzing transformers in embedding space. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16124–16170, 2023. doi: 10.18653/v1/2023.acl-long.893. URL <https://aclanthology.org/2023.acl-long.893/>.

Bradley Efron and Robert J. Tibshirani. *An Introduction to the Bootstrap*. Number 57 in *Monographs on Statistics and Applied Probability*. Chapman & Hall, New York, 1993. doi: 10.1007/978-1-4899-4541-9. URL <https://link.springer.com/book/10.1007/978-1-4899-4541-9>.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. A mathematical framework for transformer circuits. *Transformer Circuits*

Thread, 2021. URL <https://transformer-circuits.pub/2021/framework/index.html>.

Sheridan Feucht, David Atkinson, Byron C. Wallace, and David Bau. Token erasure as a footprint of implicit vocabulary items in LLMs. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9727–9739, 2024. doi: 10.18653/v1/2024.emnlp-main.543. URL <https://aclanthology.org/2024.emnlp-main.543/>.

Atticus Geiger, Hanson Lu, Thomas Icard, and Christopher Potts. Causal abstractions of neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pages 9574–9586. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/hash/4f5c422f4d49a5a807eda27434231040-Abstract.html.

Mor Geva, Avi Caciularu, Kevin Wang, and Yoav Goldberg. Transformer feed-forward layers build predictions by promoting concepts in the vocabulary space. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 30–45, 2022. doi: 10.18653/v1/2022.emnlp-main.3. URL <https://aclanthology.org/2022.emnlp-main.3/>.

Mor Geva, Jasmijn Bastings, Katja Filippova, and Amir Globerson. Dissecting recall of factual associations in autoregressive language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 12216–12235, 2023. doi: 10.18653/v1/2023.emnlp-main.751. URL <https://aclanthology.org/2023.emnlp-main.751/>.

Asma Ghandeharioun, Avi Caciularu, Adam Pearce, Lucas Dixon, and Mor Geva. Patchscopes: A unifying framework for inspecting hidden representations of language models. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pages 15466–15490. PMLR, 2024. URL <https://proceedings.mlr.press/v235/ghandeharioun24a.html>. arXiv:2401.06102.

Xijie Gong, Zimeng Huang, and Tonghan Wang. Verbalizing multi-token concepts in LLMs. *arXiv preprint arXiv:2608.31084*, 2026a. doi: 10.48550/arXiv.2608.31084. URL <https://arxiv.org/abs/2608.31084>.

Xilin Gong, Shu Yang, Zehua Cao, Lynne Billard, and Di Wang. Faithful-Patchscopes: Understanding and mitigating model bias in hidden representations explanation of large language models. *arXiv preprint arXiv:2602.00300*, 2026b. doi: 10.48550/arXiv.2602.00300. URL <https://arxiv.org/abs/2602.00300>.

Wes Gurnee, Nicholas Sofroniew, Adam Pearce, Mateusz Piotrowski, Isaac Kauvar, Runjin Chen, Anna Soligo, Paul Bogdan, Euan Ong, Rowan Wang, T. Ben Thompson, David Abrahams, Subhash Kantamneni, Emmanuel Ameisen, Joshua Batson, and Jack Lindsey. Verbalizable representations form a global workspace in language models. *Transformer Circuits Thread*, 2026. URL <https://transformer-circuits.pub/2026/workspace/index.html>. arXiv:2607.15495.

Stefan Heimersheim and Neel Nanda. How to use and interpret activation patching. *arXiv preprint arXiv:2404.15255*, 2024. doi: 10.48550/arXiv.2404.15255. URL <https://arxiv.org/abs/2404.15255>.

John Hewitt and Percy Liang. Designing and interpreting probes with control tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2733–2743, 2019. doi: 10.18653/v1/D19-1275. URL <https://aclanthology.org/D19-1275/>.

Sture Holm. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6(2):65–70, 1979. URL <https://www.jstor.org/stable/4615733>.

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 (NeurIPS)*, volume 37, pages 137903–137964. Curran Associates, Inc., 2024. doi: 10.52202/079017-4380. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/f94cfd15db3f16ee7789b6b7e91ec476-Abstract-Conference.html. arXiv:2405.17653.

Robert Huben, Hoagy Cunningham, Logan Riggs Smith, Aidan Ewart, and Lee Sharkey. Sparse autoencoders find highly interpretable features in language models. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024. URL <https://openreview.net/forum?id=F76bwRSLeK>. arXiv:2309.08600.

Guy Kaplan, Matanel Oren, Yuval Reif, and Roy Schwartz. From tokens to words: On the inner lexicon of LLMs. In *The Thirteenth International Conference on Learning Representations (ICLR)*, 2025. URL <https://openreview.net/forum?id=328vch6tRs>. arXiv:2410.05864.

Millicent Li, Alberto Mario Ceballos Arroyo, Giordano Rogers, Naomi Saphra, and Byron C. Wallace. Do activation verbalization methods convey privileged information? In *Proceedings of the 43rd International Conference on Machine Learning (ICML)*, volume 306 of *Proceedings of Machine Learning Research*, pages 66102–66142. PMLR, 2026. URL <https://proceedings.mlr.press/v306/li26k.html>. arXiv:2509.13316.

Jack Lindsey, Wes Gurnee, Emmanuel Ameisen, Brian Chen, Adam Pearce, Nicholas L. Turner, Craig Citro, David Abrahams, Shan Carter, Basil Hosmer, Jonathan Marcus, Michael Sklar, Adly Templeton, Trenton Bricken, Callum McDougall, Hoagy Cunningham, Thomas Henighan, Adam Jermyn, Andy Jones, Andrew Persic, Zhenyi Qi, T. Ben Thompson, Sam Zimmerman, Kelley Rivoire, Thomas Conerly, Chris Olah, and Joshua Batson. On the biology of a large language model. *Transformer Circuits Thread*, 2025. URL <https://transformer-circuits.pub/2025/attribution-graphs/biology.html>.

Xukai Liu, Ye Liu, Jipeng Zhang, Yanghai Zhang, Kai Zhang, and Qi Liu. Layer-order inversion: Rethinking latent multi-hop reasoning in large language models. *arXiv preprint arXiv:2601.03542*, 2026. doi: 10.48550/arXiv.2601.03542. URL <https://arxiv.org/abs/2601.03542>.

Quinn McNemar. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157, 1947. doi: 10.1007/BF02295996. URL <https://doi.org/10.1007/BF02295996>.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pages 17359–17372. Curran Associates, Inc., 2022. doi: 10.52202/068431-1262. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/6f1d43d5a82a37e89b0665b33bf3a182-Abstract-Conference.html.

Victor Morand, Josiane Mothe, and Benjamin Piwowarski. On the representations of entities in auto-regressive large language models. In *Proceedings of the 8th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 433–451, Suzhou, China, 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.blackboxnlp-1.25. URL <https://aclanthology.org/2025.blackboxnlp-1.25/>. arXiv:2510.09421.

nostalgebraist. Interpreting GPT: the logit lens. LessWrong, 31 August 2020, 2020. URL <https://www.lesswrong.com/posts/AcKRB8wDpdaN6v6ru/interpreting-gpt-the-logit-lens>.

Koyena Pal, Jiuding Sun, Andrew Yuan, Byron C. Wallace, and David Bau. Future lens: Anticipating subsequent tokens from a single hidden state. In *Proceedings of the 27th Conference on Computational Natural Language Learning (CoNLL)*, pages 548–560, 2023. doi: 10.18653/v1/2023.conll-1.37. URL <https://aclanthology.org/2023.conll-1.37/>. arXiv:2311.04897.

Alexander Pan, Lijie Chen, and Jacob Steinhardt. LatentQA: Teaching LLMs to decode activations into natural language. In *The Fourteenth International Conference on Learning Representations (ICLR)*, 2026. URL <https://openreview.net/forum?id=niUroX9EOd>. arXiv:2412.08686.

Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, Craig Citro, Emmanuel Ameisen, Andy Jones, Hoagy Cunningham, Nicholas L. Turner, Callum McDougall, Monte MacDiarmid, Alex Tamkin, Esin Durmus, Tristan Hume, Francesco Mosconi, C. Daniel Freeman, Theodore R. Sumers, Edward Rees, Joshua Batson, Adam Jermy, Shan Carter, Chris Olah, and Tom Henighan. Scaling monosemanticity: Extracting interpretable features from Claude 3 Sonnet. Transformer Circuits Thread, 2024. URL <https://transformer-circuits.pub/2024/scaling-monosemanticity/index.html>.

Jesse Vig, Sebastian Gehrmann, Yonatan Belinkov, Sharon Qian, Daniel Nevo, Yaron Singer, and Stuart Shieber. Investigating gender bias in language models using causal mediation analysis. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 12388–12401. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/hash/92650b2e92217715fe312e6fa7b90d82-Abstract.html.

Edwin B. Wilson. Probable inference, the law of succession, and statistical inference. *Journal of the American Statistical Association*, 22(158):209–212, 1927. doi:

10.1080/01621459.1927.10502953. URL <https://doi.org/10.1080/01621459.1927.10502953>.

Wilson Wu, John X. Morris, and Lionel Levine. Do language models plan ahead for future tokens? In *First Conference on Language Modeling (COLM)*, 2024. URL <https://openreview.net/forum?id=BaOAvPUyBO>. arXiv:2404.00859.

Shi-Qi Yan, Kai-Xuan Ding, Chao-Hong Tan, Qian Chen, Wen Wang, Xiangang Li, and Zhen-Hua Ling. Short horizons and sparse concepts: a mathematical view of the readout in the J-lens. *arXiv preprint arXiv:2608.25347*, 2026. doi: 10.48550/arXiv.2608.25347. URL <https://arxiv.org/abs/2608.25347>.

An Yang, Anfeng Li, Baosong Yang, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. doi: 10.48550/arXiv.2505.09388. URL <https://arxiv.org/abs/2505.09388>.

Sohee Yang, Elena Gribovskaya, Nora Kassner, Mor Geva, and Sebastian Riedel. Do large language models latently perform multi-hop reasoning? In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10210–10229, 2024. doi: 10.18653/v1/2024.acl-long.550. URL <https://aclanthology.org/2024.acl-long.550/>. arXiv:2402.16837.

Alexander Yom Din, Taelin Karidi, Leshem Choshen, and Mor Geva. Jump to conclusions: Short-cutting transformers with linear transformations. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 9615–9625, 2024. doi: 10.63317/5griomztwysx. URL <https://aclanthology.org/2024.lrec-main.840/>.

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 (ICLR)*, 2024. URL <https://openreview.net/forum?id=Hf17y6u9BC>. arXiv:2309.16042.

Technical Appendix

In this appendix, we give the details needed to reproduce and assess the results of the main text. Section A covers the models, target prompts, and scoring, and Sections B and C describe the item sets and the controls. The remaining sections contain the full tables and secondary results (Section D), qualitative examples (Section E), and compute, code, and data (Section F). A Patchscope (Ghandeharioun et al. 2024) takes a hidden state h_ℓ from a source prompt, patches it into the residual stream of a target prompt, and lets the model generate. The Jacobian lens (J-lens) (Gurnee et al. 2026) and the logit lens (nostalgebraist 2020) are token lenses, which map a state to a ranked list of single tokens. WorkspaceBench (Blank et al. 2026b) supplies the benchmark items, model, and scorer. All intervals we report are 95% intervals, computed for rates by an item bootstrap with 1,000 resamples (Efron and Tibshirani 1993) and for paired differences by a bootstrap over answer strings with 10,000 resamples. For

benchmark counts we use the Wilson score method instead (Wilson 1927).

A Models, Target Prompts, and Scoring

Models. We use four models of the Qwen3 family (Yang et al. 2025): Qwen3-1.7B (28 layers; source layers 22 and 25), Qwen3-8B (36 layers; source layers 29 and 32), Qwen3-14B (40 layers; source layers 32 and 36), and Qwen3.6-27B (64 layers; source layers 52 and 56 for the Patchscopes and 40, 48, 52, and 56 for the token lenses). Layers are indexed from zero, so that L_k denotes the residual stream after layer k , and $L_{32} \rightarrow L_4$ denotes a state taken at layer 32 and patched into the target prompt after layer 4. Qwen3-1.7B runs in fp32. Qwen3-8B and Qwen3-14B run with bf16 weights and fp32 activations in a minimal re-implementation of the dense Qwen3 layer, and Qwen3.6-27B runs in bf16 through a hooks-based backend. The two backends agree on Qwen3-14B, the only model on which we compared them.

Target prompts. Table A1 lists both target prompts verbatim. The copy prompt has eight variants. The patched state replaces the residual stream at “?”, eight tokens are decoded greedily, and the generation is truncated at the first “;”, “->”, or newline. The continuation prompt also has eight variants: four prose prefixes that end mid-clause and four answer frames. The patched state replaces the residual stream at the final token, and twelve tokens are decoded greedily without truncation. Since an answer string in a target prompt would inflate accuracy, the code checks the prompts when it loads the item files. A copy-prompt exemplar fails this check if it equals or contains any item string, answer, alias, required answer form, bridge answer, or misspelled word in our item files and the six benchmark banks. The continuation prompt and two further prompts used on the benchmark (a copy prompt with longer exemplars and a spelling-correction prompt) are also checked in the reverse direction, which rejects a prompt if an item string of three or more characters appears in its text as a whole word. We left the reverse check out for the copy-prompt exemplars and instead checked the new item file for overlap with all prompts when we built it (Section B). We added the overlap check after an earlier prompt set, in which 21 of the 24 exemplars overlapped with item strings, inflated two-word accuracy by 0.07. All reported runs use the prompt set that replaced it.

Scoring on our items. Generation is greedy for a fixed number of tokens and ignores end-of-sequence tokens. Before comparison, each generation is stripped, internal whitespace is collapsed, and case is folded. Under the *exact rule*, a prompt variant is correct when it generates the exact normalized answer. The *majority* rate counts an item as correct when more than four of the eight prompt variants are correct, and the *any-variant* rate when at least one is. Because the continuation prompt’s output is not cut at a delimiter, we score it on our items by whether the answer is a string prefix of its output. This rule also credits an answer that is a prefix of a longer alphanumeric string (e.g., “North Korea” in “North Korean”). Token lenses are read with the cosine readout of the benchmark, and their ten highest-ranked tokens are scored as ten samples.

Scoring on the benchmark. All methods are scored by the benchmark’s regular expression with its any-layer rule. Each required answer component must match at word boundaries, after Unicode folding, inside some generated sample at one source layer, and different components may match different samples of that layer. Because a multi-token answer form cannot equal a single vocabulary token, both token lenses score 0.00 on every multi-token family by construction. The benchmark reports token-lens results after a large language model (LLM) summarizes the token list into a sentence. Of the three denominators we report, *all* counts the 100 items of a family. An item is *answerable* when the case-folded greedy continuation of the plain prompt contains the answer or an alias within 48 tokens, and *immediately answered* when that continuation begins with the answer after an optional article. This answerability filter uses the plain prompt and differs from the chat-format filter of the benchmark.

B Item Sets

Two-word and sub-word items. Each item is a cloze-style prompt with a multi-token answer. The 609 two-word items have answers that name people, places, compounds, or events, and the 867 sub-word items have answers that the tokenizer splits into multiple pieces, such as rare words, numbers, and translations. Each group is divided into a tuning half and a held-out half. All main results use held-out items that the model answers correctly under greedy decoding.

Bridge items. Both groups also contain two-hop *bridge* items such as “Fact: The continent of the country whose capital is Riyadh is”, whose intermediate entity (“Saudi Arabia”) is the string to decode. The first-hop token is the last token of the phrase that names the first-hop entity (“Riyadh”). The 30 items for which we located a first-hop token share 17 distinct cues, because several items differ only in the relation asked.

New test set. We built a new test set of 1,002 items whose answers do not overlap with material used in earlier runs. We removed a candidate if its answer equaled an earlier item string, a benchmark answer form, or a prompt exemplar. We also removed it if its answer contained, or was contained in, any such string that is name-like (capitalized or multi-word). Containment was tested at whole words, and lowercase single nouns were removed only on exact equality. This filter removed 810 candidates.

Benchmark items. WorkspaceBench provides 100 items per family (basic readout, typo correction, and multihop) at a fixed commit and reads the final prompt token. Table A2 gives the number of items in each reported condition.

C Controls

Substitute states. We evaluate every reported condition with substitute states. Analogous to control tasks for probes (Hewitt and Liang 2019), these measure the accuracy that the target prompt or the item’s category produces without the item’s own state. The *shuffled* control assigns each item the state of another answerable item of the same category, through a fixed-seed derangement subject to the condition

copy prompt, eight variants (the state replaces the residual stream at “?”; eight greedy tokens, truncated at “;”, “->”, or newline)

Grand Rapids -> Grand Rapids; kaleidoscope -> kaleidoscope; 3817 -> 3817; ?
Chiang Mai -> Chiang Mai; serendipity -> serendipity; 5926 -> 5926; ?
Alice Springs -> Alice Springs; ventriloquist -> ventriloquist; 7403 -> 7403; ?
Milton Keynes -> Milton Keynes; gobbledygook -> gobbledygook; 6158 -> 6158; ?
Punta Arenas -> Punta Arenas; flabbergasted -> flabbergasted; 4271 -> 4271; ?
Ann Arbor -> Ann Arbor; pomegranate -> pomegranate; 8539 -> 8539; ?
Boca Raton -> Boca Raton; Schadenfreude -> Schadenfreude; 2764 -> 2764; ?
Coffs Harbour -> Coffs Harbour; paraphernalia -> paraphernalia; 9385 -> 9385; ?

continuation prompt, eight variants (the state replaces the residual stream at the final token; twelve greedy tokens, not truncated; prose prefixes first, answer frames last)

The committee met on Tuesday to review the proposal, and after a short discussion the
chair announced that the
According to the report published last spring, the most important factor turned out to be
the
She opened the letter slowly and read the first line, which mentioned the
In the second chapter the author turns to the question of the
Question: what is it called?\nAnswer: the
The answer is
Q: What is its name?\nA:
The correct term for it is

Table A1: The two target prompts. Each line is one of the eight prompt variants used for majority voting.

item set	model	items
two-word, held-out	14B	175
	8B	160
sub-word, held-out	14B	342
	8B	340
	1.7B	278
two-word, new	14B	175
sub-word, new	14B	431
single-token answers	14B	128
bridge, last token	14B	143
bridge, first-hop token	14B	30 (17 cues)
new bridge	14B	123
benchmark basic	27B	85 / 51
	14B	63 / 28
	8B	49 / 25
benchmark typo	27B	34 / 0
	14B	25 / 0
benchmark multihop	27B	86 / 83
	14B	66 / 60

Table A2: Number of items per condition. For our item sets, the counts include only held-out or new items that the model answers correctly. For WorkspaceBench, which has 100 items per task, each entry gives the number of answerable / immediately answered items.

that paired items have different answers. We drop items without a valid partner, which removes 4 of 175 two-word and 3 of 342 sub-word items on Qwen3-14B. The *cross-category* control pairs items across categories under the same rule. On the benchmark, 35 of 63 answerable basic items and 9 of 25 typo items lack a cross-category partner. We compute this control on the remaining items and report its denominator wherever we quote it. The *random* control patches a Gaussian vector scaled to the norm of the item’s own state at the same layer and position. On the benchmark, a fourth control decodes every prompt variant without a patch and scores those samples for every item. On their own, the continuation prefixes produce generic prose such as “ committee would meet again on Thursday” and the copy prompts produce a blank, so the unpatched generations contain no answers. A control with zero successes bounds the false-positive rate only loosely, with two-sided 95% Wilson upper bounds of .12 for 0 of 28 and .06 for 0 of 63.

Control positions. In earlier runs, the shuffled and cross-category states were read at the original item’s final-token index within the partner’s prompt. That index falls before the partner’s final token whenever the partner prompt is longer, which holds for 70 of 171 pairs in the two-word set. We corrected the controls to read the partner’s own final token, or its own first-hop token for bridge items. This is the one correction we made to the controls. We consider a condition invalid if its corrected control exceeds .05. The corrected controls are at most .007 by majority and .011 by any-variant, and the conclusions are the same as before the correction. The real-state and random-vector conditions were unaffected.

D Additional Results

D.1 Exact Recovery with an Early Target Layer

Table A3 compares the same-layer and the early-layer Patchscope on three models and on the new items. It also gives the paired improvement and its interval from a bootstrap over answer strings (10,000 resamples, seed 0), which treats repeated answers as one unit. On the original two-word set, 52 items change from incorrect to correct and 6 change in the opposite direction, and in every row the clustered interval is close to the item-level interval. On Qwen3-8B, the copy prompt patched at layer 4 reproduces its own exemplars (for example, “d -> d; 123”) and fails on every item. Given this result, we then evaluated layer 8, one of the candidate target layers, and adopted it as the target layer for Qwen3-8B.

model, items	<i>n</i>	same	early [95% CI]	Δ [CI]
14B two-word (orig.)	175	.206	.469 [.39,.54]	.26 [.18,.34]
14B two-word (new)	175	.086	.337 [.27,.41]	.25 [.18,.33]
14B sub-word (orig.)	342	.129	.310 [.26,.36]	.18 [.13,.24]
14B sub-word (new)	431	.044	.169 [.14,.20]	.13 [.09,.16]
8B two-word	160	.250	.494 [.41,.57]	.24 [.15,.34]
8B sub-word	340	.168	.379 [.33,.43]	.21 [.16,.27]
1.7B sub-word	278	.115	.219 [.17,.27]	.10 [.06,.15]

Table A3: Exact-match accuracy (majority rate) of the same-layer and early-layer Patchscope. Columns give the number of items (*n*), the accuracy at the source layer (same), the accuracy at the early target layer (early) with a 95% interval, and the paired improvement Δ (early minus same) with a 95% interval from a bootstrap over answer strings. Source layers lie at about 80% of depth (14B: L32, 8B: L29, 1.7B: L22). The target layer is 4 for the 14B, 8 for the 8B (at layer 4 the copy prompt reproduces its exemplars), and 8 for the 1.7B (the best of its candidate layers). Control accuracy is at most .000 for the 14B, .003 for the 8B, and .007 for the 1.7B. Bold marks accuracy significantly above 30% (one-sided exact binomial test, Holm-adjusted; Holm 1979); the 14B sub-word and new two-word rates exceed 30% by point estimate only.

We characterize the effect with two further analyses, and in the first we split the items post hoc by the rank of the correct second token in an unpatched prompt given only the first token. On the 96 hard two-word items (rank above 10), the same-layer Patchscope achieves .021 and the early-layer Patchscope .344. On the 220 hard sub-word items, the rates are .041 and .232. The shuffled control is 0 on every subset. While this split rules out one explanation, completion from the first token, reconstruction from entity and relation cues remains possible, and so we interpret exact recovery as decodability. In the second analysis, we replace the exact rule with the benchmark’s word-boundary matcher, and the 14B L32 $\rightarrow$ L4 rates become .480 for two-word and .404 for sub-word items by majority, with every control at or below .008. The exact rule undercounts sub-word answers by about 0.1, because a generation such as “giraffe is” fails the exact rule and passes the benchmark matcher.

D.2 Linear and First-Order Readouts

Table A4 summarizes all readouts of the second answer token at layer 32 of Qwen3-14B. The step readout extends the J-lens by one position with a first-order approximation of the model. Given the state and the true first token, it ranks the correct second token first for at most one item at any layer on either Qwen3-1.7B or Qwen3-14B, and only for items where the lens itself ranks the first token outside its top ten. We also built a beam-search decoder that applies the step readout token by token and configured it on the tuning half. On the held-out half, its exact recall is 0 at both 1.7B layers (60 items at L18, 278 at L22) under every stopping rule. The generic-text linear probe performs at or near its shuffled control on sub-word answers at every layer we evaluated (24, 32, and 36 on the 14B; 14 to 26 on the 1.7B). It exceeds its control only on two-word answers, where the paired difference at layer 32 is .047 [.018, .082] (exact McNemar test, McNemar 1947; $p = .008$). The same probe predicts the first answer token with up to .61 top-1 accuracy on the 14B and .90 on the 1.7B, and we take this as evidence that the probe is fitted correctly.

readout	statistic	value	control
step readout*	2nd top-1	1/285	0 if 1st in top-10
beam decoder, 1.7B	recall@10	0/60, 0/278	(L18, L22)
probe, sub-word	2nd top-10	.149	shuffled .150
probe, two-word	2nd top-10	.046	shuffled .000
J-lens, two-word	2nd top-10	.434	
probe, single-token	1st top-1	.23-.61	(L32, L36)
closed list of 488	top-1	.777	chance .002
patch*, replace	2nd top-10	.395	shuffled .053
patch*, add	2nd top-10	.436	shuffled .127
tangent of add*	2nd top-10	.073	shuffled .027

Table A4: Readouts of the second answer token (higher is better). Unless a row states otherwise, the model is Qwen3-14B and the layer is 32. *1st* and *2nd* denote the first and second answer tokens. *Top-k* means that the correct token is among the *k* highest-ranked tokens, and *recall@10* that the exact answer is among the decoder’s ten outputs. Rows marked * receive the true first token. The closed-list row ranks 488 candidate answers by their J-lens scores. It is a diagnostic and does not generate text. The patch rows insert h_ℓ into a generic prompt one position before the first token, by replacement or by addition, and the tangent row is the first-order approximation of the additive patch. The last column gives a reference value for each row: a control state, chance level, the layers evaluated, or, for the step readout, its value on the items where the lens ranks the first token in its top ten.

To relate this pattern to the lens itself, we split the vector that the J-lens assigns to each vocabulary token into a term that predicts the current position and terms that predict later positions. The ratio of the same-position norm to the norm of all future terms grows with depth, from 0.6 to 3.7 on the 1.7B and from 0.8 to 2.4 on the 14B (about 40% to 80% of depth). Within the future terms, the next position accounts for only 7 to 15%, and positions twenty or more tokens ahead account for 40 to 55%. The second answer token lies one position

ahead, where the lens places the least weight, in line with the step-readout and probe results above. The one-position patch in Table A4 inserts the state into 64 generic prompts one position before the first answer token, appends the true first token, and ranks the correct second token by the change in its log-probability averaged over prompts. The tangent row is the exact directional derivative of the additive map along the state, that is, the first-order counterpart of the additive patch.

D.3 Patching Every Lower Layer

We consider two hypotheses for the low accuracy of the same-layer Patchscope. Under the first, placeholder contamination (Ghandeharioun et al. 2024, App. C), the representation of the placeholder token persists in the layers below the patch and the generation attends to it. Under the second, a state patched at layer 4 passes through 28 further layers before the source layer, and these layers perform computation that the answer requires. To distinguish the two, we patch the state at the patch position into every layer up to k , at the embedding and after each layer, which removes the representation of the placeholder token from all layers below k . This intervention also fixes the trajectory of the patched position, rewrites the lower-layer keys and values that later positions attend to, and presents the state repeatedly. Table A5 gives the counts and paired differences behind Figure 3.

	copy prompt	continuation
same layer	36	66
every layer ≤ 32	61	120
every layer ≤ 4	83	129
layer 4 only	82	133
all-32 – same	.143 [.086, .203]	.309 [.222, .395]
layer 4 – all-32	.120 [.046, .196]	.074 [–.011, .162]
all-4 – layer 4	.006 [–.012, .028]	–.023 [–.057, .006]
layer 4 – same	.263 [.183, .343]	.383 [.295, .474]
gap share, all-32	.54 [.35, .78]	.81 [.62, 1.02]

Table A5: Patching every lower layer (Qwen3-14B, source layer 32, 175 two-word items). Top: number of correct items in each condition, where all- k patches every layer up to k . Middle: paired differences between conditions, with 95% intervals from a bootstrap over answer strings. Bottom: fraction of the early-layer improvement that all-32 recovers, with a 95% item-bootstrap interval.

We observe an intermediate outcome when the state is patched into every layer up to the source layer. Copy-prompt accuracy increases from 0.206 to 0.349, which closes 54% of the gap to the single patch at layer 4 (0.469; item-bootstrap interval 35% to 78%). For the continuation prompt, the same intervention increases accuracy from 0.377 to 0.686 and closes 81% of its gap (62% to 102%). Patching every layer up to 4 matches the single patch at layer 4 for both prompts (83 vs. 82 and 129 vs. 133 of 175 items; Table A5 gives the intervals). Once the state is patched into layers 0 to 4, the representation of the placeholder token in those layers has no measurable effect. The all-layer condition lies more than .05 from both the single patch at layer 4, the outcome

that the first hypothesis predicts, and the same-layer patch, the outcome that the second predicts. The gap shares are descriptive ratios of intervention outcomes and should not be read as causal fractions. Because the all-layer condition changes more than the placeholder representation, the test does not separate contamination from computation. One observation about the continuation prompt bears on the first hypothesis without settling it. Even though this prompt ends in an ordinary word and lacks a placeholder, the all-layer condition closes a larger share of its gap. The deficit of the same-layer Patchscope is not specific to the “?” token.

D.4 WorkspaceBench

Table A6 reports all methods on the benchmark’s model over all 100 items of each family. In the default configuration, L52 $\rightarrow$ L8, the continuation prompt is correct on 14 of 100 basic-readout items, which are 14 of the 85 answerable and 11 of the 51 immediately answered items. We select layer 4 as the best of three target layers on the basic-readout items themselves, counting an item as correct if either of two source layers recovers it, and this configuration is correct on 22 of them. At this configuration, typo correction and multihop reach .33 and .06. All raw token lenses score 0.00, and the shuffled control is 0 in every family. The published token-lens results summarize a list of tokens with an LLM before scoring, and our results score raw generations, so a similar score does not mean that the two methods are equivalent.

The choice of prompt variant accounts for most of the benchmark result. Of the 22 correct basic-readout items in the either-layer configuration, the four answer frames recover all 22 and the four prose prefixes together recover 12. On typo correction, the answer frames recover 33 items and the prose prefixes 1, and the single frame “The correct term for it is” recovers 29 of the 33. Because every unpatched generation is incorrect (Section C), we can rule out leakage from the prompt text. An answer frame nonetheless acts as a task instruction, which the copy prompt lacks, and the copy prompt fails on Qwen3.6-27B at every target layer (at most .06 basic and .09 typo without truncation).

The 14B pilot rests on small counts, so its intervals are wide. At L32 $\rightarrow$ L4, the continuation prompt recovers 9 of 63 answerable items (.143, Wilson interval [.08, .25]) and 8 of 28 immediately answered items (.286, Wilson interval [.15, .47]). The four controls in this configuration are all zero, with the cross-category control computed on the 28 items that have a partner. Three of the nine correct items match only an alias of the answer (.095 answerable without aliases). The later all-100 run reproduced these counts exactly, and because decoding is greedy, this agreement checks determinism only.

D.5 Bridge Entities

Table A7 lists every bridge readout and position on Qwen3-14B at layer 32. Under the exact rule, the copy prompt almost never recovers the bridge entity at the final token of a two-hop prompt, and it recovers 0 of 30 at the first-hop token. The continuation prompt at the first-hop token names the bridge entity, the country of the first-hop entity, for 26 of 30 items under the benchmark rule. A neutral-context control accounts for most of this effect, since the same 30 entities

method on Qwen3.6-27B, all 100 items	basic	typo	multihop
<i>ours (training-free, no summarizer)</i>			
continuation, L52 → L8 (default)	.14	.21	.02
continuation, layer 4, either of L52 and L56 [†]	.22	.33	.06
continuation, same layer	.06	.00	.01
copy prompt, layer 4 (without truncation in parentheses)	.04 (.06)	.00 (.09)	.01 (.01)
J-lens, raw	.00	.00	.00
logit lens, raw	.00	.00	.00
shuffled control (either-layer configuration)	.00	.00	.00
<i>published on the same model and items</i>			
J-lens, raw	.00	.00	.00
J-lens, LLM-summarized	.24	.17	.08
logit lens, LLM-summarized	.20	.20	.08
natural-language autoencoder (trained)	.53	.75	.22
oracle lens (trained)	.71	.83	.32
prompt-only (sees the prompt, no activation)	.59	.67	.69

Table A6: WorkspaceBench pass rate on Qwen3.6-27B over all 100 items of each task (higher is better). [†]Best of three target layers, counting an item as correct if either source layer (L52 or L56) recovers it. On basic readout, L52 alone gives .17 and L56 alone .15. Requiring all answer components within one sample gives .21, and counting only the primary answer string without aliases gives .16. On typo correction, 26 of the 33 correct items are words that the model itself does not correct within 48 tokens.

placed in two neutral sentences that do not ask about their country (“Fact: Yesterday someone mentioned Riyadh is”) yield 43 of 60, a rate of .717. The paired difference between the two-hop and the neutral prompt is .200 on one template and .100 on the other, with 95% intervals of [.032, .414] and [.000, .267] when clustered over the 17 distinct first-hop cues. Our interpretation is that the first-hop state contains enough information about the entity for the Patchscope to name its country, and that the two-hop context adds at most a small, template-dependent increment. A Patchscope that recomputes the country from the entity would produce this pattern. The pattern does not establish that reading, and it leaves open whether the source state also contains bridge information. The new bridge items yield an exact rate of 0 for every method and position. Their continuation-prompt rate under the benchmark rule, .203 [.138, .276], is measured at the end of the first-hop description and is not directly comparable with the original bridge items. On the multihop items of the benchmark, we scanned the last 40 prompt positions on Qwen3-14B (66 answerable items, two of the four answer frames, L32 → L4). The maximum bridge rate is .039 (position 35 from the end, $n = 51$; shuffled maximum 0), and the final answer reaches .258 at the last token. On Qwen3.6-27B, read at the last token only, the bridge rate reaches .06 and the final-answer rate .35.

E Qualitative Examples

All examples are drawn at random from the sorted list of item identifiers with a fixed seed of 0. Table A8 shows eight held-out two-word items on Qwen3-14B at layer 32 with the copy prompt. At layer 4, three items are recovered by all eight variants (two of these three are not recovered at the source layer), and four items fail in both configurations. Six new sub-word items at layer 4 show a similar mix, with “Degas” generated by 4 of 8 variants (otherwise “Dali” or “Dancer”), “Pollock” by 1 of 8 (“Warhol” three times), and

“Holz” by 2 of 8 (“Woods”, “Walden”). “paz” is matched by only 1 of 8 variants under the exact rule, because seven variants generate “paz” → “paz;” and the quotation marks prevent a match. “141” and “sixty-five” are each matched by 0 of 8 variants, because the variants reproduce their own numbers (for “141”, “1234 → 12”).

On the benchmark, six basic-readout items on Qwen3.6-27B with the continuation prompt at L52 → L4 illustrate what the benchmark rule credits. The lowest point of Australia (answer “Kati Thanda-Lake Eyre”) counts as correct, although its samples also include “the Great Salt Lake”, and Lake Turkana counts as correct (“Lake Turkana. The chapter begins with a d”). The Polish “Ziemie Odzyskane” (“Ziemia Pruska” twice), McCune-Albright syndrome (“Tuberculosis”, “Amino acid”), and Thabana Ntlenyana (“the highest mountain in Africa. A. Kilima”) are incorrect. Nasir al-Din al-Tusi, whose prompt and answer form are in Arabic, is incorrect although five of eight samples generate “Nasir al-Din al-Tusi” in Latin script.

We include two further items in full, the second of which, the Riyadh item, also appears in the main paper. For “The national monetary unit of the Himalayan kingdom governed from Thimphu is the” (answer “Bhutanese ngultrum”, alias “ngultrum”), the Qwen3.6-27B samples at L52 → L4 include “Ngultrum (Nu.) would be the official currency of”, “ngultrum. The ngultrum is the currency of”, and “Nu. The currency of Bhutan is the Ngultrum”. The shuffled control, which receives another item’s state, generates “the Hong Kong dollar” and “the Macau Pataca (MOP)”. In the same configuration, the copy prompt generates a blank six times and “Ngultrum” twice, and the J-lens top ten consists of underscores and “Tibetan”. For “Fact: The continent of the country whose capital is Riyadh is”, read at “Riyadh” on Qwen3-14B, the continuation samples include “is the capital of Saudi Arabia. What” and “is “the Kingdom of Saudi Arabia””. The shuffled control, given the state of the Free-

method	position	n	exact majority	exact any-variant	benchmark any-variant [95% CI]
copy, same layer	final token	143	.007	.007	
copy, layer 4	final token	143	.028	.049	
copy	first-hop token	30	0	0	.133
continuation, layer 4	first-hop token	30			.867 [.733, .967]
continuation, layer 4	first-hop token +1	30			.167
continuation, layer 4	first-hop token -1	30			.000
continuation, layer 4	final token	143			.196
continuation, shuffled / random	first-hop token	23 / 30			0 / 0
continuation, neutral sentence	entity token	60			.717 [.600, .817]
copy, neutral sentence	entity token	60			.500
new bridges, every method	every position	123	0		
new bridges, continuation	end of description	123			.203 [.138, .276]

Table A7: Bridge-entity recovery on two-hop items (Qwen3-14B, layer 32). The two exact columns count an item as correct under the exact rule, by majority or by any variant. The last column uses the benchmark’s matcher with any-variant counting. The neutral-sentence rows place the same 30 entities in two templates (20 of 30 and 23 of 30 items correct). New bridge items are read at the end of the first-hop description.

answer	same layer	layer 4
Edward Jenner	0/8 (Edward)	8/8
Isaac Newton	2/8 (Isaac)	2/8 (Isaac)
one hundred	0/8 (100)	0/8 (100)
Marilyn Monroe	2/8 (born in 1980)	0/8 (born)
United Kingdom	5/8	8/8
Nova Scotia	7/8	6/8
Johannes Kepler	0/8 (Johannes)	0/8 (Kepler)
Mother Teresa	0/8 (Mother)	8/8

Table A8: Randomly sampled held-out two-word items (Qwen3-14B, source layer 32, copy prompt). Each entry gives the number of variants that generate the exact answer and, in parentheses, the most frequent output when it differs from the answer. Bold marks a majority.

town item, generates “ is the capital of Sierra Leone”, and the neutral prompt “Fact: Yesterday someone mentioned Riyadh is” yields “ , the capital of Saudi Arabia, is” for six of eight variants.

F Compute, Code, and Data

Code and data. All code (model backends, Patchscopes, controls, item construction, and the analysis scripts that produce every table), the item files, and every per-item result file are available at <https://github.com/deadmash07/multitoken-readout> under the MIT license. The repository also contains a provenance file (PROVENANCE.md) that maps every reported number to its result file, configuration, target prompt, token budget, scoring rule, and answerability filter. The same file gives the recomputed numerator and denominator, the coverage of each control, and the tables broken down by layer, variant, and denominator that this appendix summarizes.

Environment. GPU runs used one NVIDIA H100 80 GB or A100-SXM4 80 GB per job on Linux, with Python 3.11, PyTorch 2.8.0 (CUDA 12.8), and transformers 5.17.0. Decoding is greedy and generations are deterministic given the

weights. Every random element (derangements for the shuffled control, Gaussian vectors for the random control, and bootstrap resamples) uses a fixed generator seed, 0 by default and recorded in each result file. The linear probe selects its ridge penalty by 5-fold cross-validation on the training split over $\{10^{-3}, 10^{-2}, 10^{-1}, 1, 10\}$, and the selected value is stored with each run. We configured the beam decoder on the tuning half (beam width 10, at most 5 tokens, and a conformal stopping rule; the full configuration is in the repository) and evaluated it on the held-out half.

Compute. The lens decomposition, second-token, and beam experiments on Qwen3-1.7B and Qwen3-14B used about 75 H100-hours. The reported Patchscope, probe, control, benchmark, lower-layer patching, bridge, and new-item runs on Qwen3-8B, Qwen3-14B, and Qwen3.6-27B used about 215 A100-hours in total, according to the wall-clock times recorded in the result files. Each Qwen3.6-27B run took 7.2 to 7.6 hours on one A100. About 140 A100-hours of earlier runs were discarded after the prompt-overlap problem described in Section A and are not reported. The bootstrap intervals, rescoring under the benchmark rule, primary-answer recount, and paired and clustered analyses ran on CPU.