Additive Relational Bindings in Transformers: What Sparse Autoencoders Miss

Sebastian Hoenig [*1] **Su Ji Park** [*1] **Kushal Jain** [1] **Bart Bussmann** [1] **Patrick Leask** [2]

Abstract

Language models often need to represent which entities are bound to which attributes, as in "Alice lives in Paris. Bob lives in London." How models construct such binding representations is poorly understood, and it remains unclear whether sparse autoencoders (SAEs) recover the binding representations that models actually use. We train a 2-layer attention-only transformer on a synthetic relational retrieval task and reverse-engineer the circuit that solves the task perfectly. We find that Layer 0 writes an approximately additive entity–relation address and a separate payload at each fact slot, while Layer 1 retrieves the matching payload by same-head query-key matching against these addresses. Linear probes decode the joint address with 100% accuracy, an additive decomposition explains 99.8% of its variance, and causal patches over the address flip predictions to a distractor. However, SAEs trained on the same activation site do not recover the joint address as clean individual features, despite reconstructions preserving full task accuracy. This provides a concrete example of a composed representation that is linearly decodable and causally used, yet not cleanly exposed as sparse features by an SAE.

1. Introduction

Language models often face tasks where correctness depends on preserving bindings between variables, not merely representing the variables themselves. In a context such as "Alice lives in Paris. Bob lives in London," the model must keep track of which city belongs to which person. This is an instance of relational composition: multiple pieces of information must be combined into a structured representation that can later be queried. Wattenberg & Viégas (2024) argue that such composition mechanisms are still poorly understood and that different forms of relational composition have different implications for feature-based interpretability methods such as Sparse Autoencoders (SAEs).

Prior work provides evidence that language models maintain entity bindings in context. Feng & Steinhardt (2023) identify binding-ID-like directions used to associate entities with attributes across several model families, and Prakash et al. (2024) trace entity-tracking mechanisms in pretrained and fine-tuned models. Recent work on multi-entity "slots" further shows that models can represent multiple entities at a single token position, while emphasizing a distinction between information that is linearly available and information the model functionally uses (Bogdan & Lindsey, 2026). These results motivate a narrower mechanistic question: when a model uses an address-like representation to retrieve a bound value, how is that address constructed, how is it consumed by the circuit, and can it be recovered by sparse feature methods?

We study this question in a controlled toy transformer trained on a synthetic relational retrieval task. Toy models have repeatedly been informative for mechanisms later studied at scale, as in modular arithmetic (Nanda et al., 2023; Baeumel et al., 2025) and superposition (Elhage et al., 2022; Bricken et al., 2023). More closely related, recent toy-model work on relative-magnitude relational composition shows that attention-only transformers can learn nontrivial composition mechanisms with implications for the common view of SAE latents as binary, independent features (Farrell et al., 2025). Our setting targets a different mechanism. Instead of ordered list copying, the model must retrieve a payload entity by constructing and matching an entity–relation address.

We make three contributions. First, we train a 2-layer attention-only transformer on a synthetic relational retrieval task and identify the circuit that solves it. Second, we show that one Layer 0 attention head represents each fact address as the sum of an entity component and a relation component, while Layer 1 uses this address to match the query to the correct fact slot. Third, we find a gap between linear decodability and SAE feature recovery: the composed address is perfectly linearly decodable and causally controls retrieval, yet the SAE learns no clean latent or group of latents for it. We trace this gap to the additivity of the

[1]Independent [2]Department of Computer Science, Durham University. Correspondence to: Sebastian Hoenig <hoenigsebastian@gmail.com>, Su Ji Park <sp3581@columbia.edu>.

Accepted to the 2nd Workshop on Compositional Learning at ICML 2026, Seoul, South Korea. Copyright 2026 by the author(s).

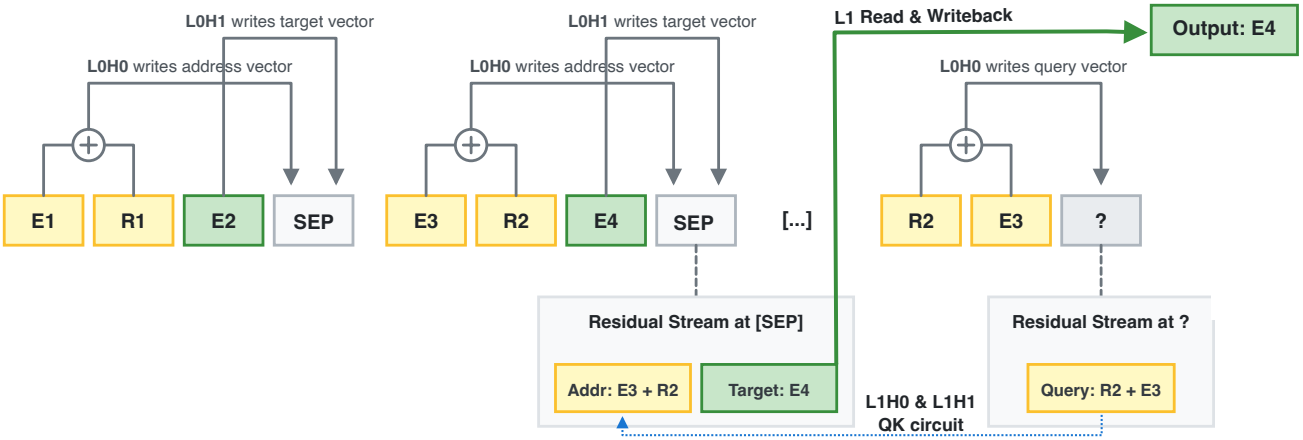

Figure 1. Staged retrieval circuit. At each fact, the first head of Layer 0 (L0H0) writes an address representation (E_1, R), the second (L0H1) a payload representation E_2 to the $[SEP]$ position. At the query, L0H0 writes the queried address (Eq, Rq) to the [?] position. The two Layer 1 heads (L1H0 and L1H1) then match the query against fact addresses via their QK circuits and retrieve the corresponding payload.

representation: because an additive address is already reconstructible from its marginal directions, no dedicated latent emerges, and the binding is carried in how strongly latents fire, not in which ones turn on. Dataset, code, and model checkpoints are available at: https://github.com/sebastianhoenig/relational-composition.

2. Methods

2.1. Model

We train a 2-layer, 2-head attention-only transformer with residual-stream dimension $d_{\text{model}} = 256$ and head dimension $d_{\text{head}} = 128$. The model uses Rotary Positional Embeddings (RoPE) and layer normalization. To force targeted entity retrieval, the output vocabulary is restricted to the 100 entity tokens.

2.2. Task

We study a synthetic relational retrieval task designed to isolate entity–relation binding. Let $\mathcal{E}$ be a set of 100 entity tokens and $\mathcal{R}$ a set of 10 relation tokens. We additionally use two dedicated synthetic tokens: a separator token `[SEP]` and a query marker `[?]`. Each prompt contains $n \in \{4, \ldots, 8\}$ facts, serialized as

$$E_{1,i}\ R_i\ E_{2,i}\ \texttt{[SEP]},$$

where $E_{1,i}, E_{2,i} \in \mathcal{E}$ and $R_i \in \mathcal{R}$. The prompt ends with a query

$$R_q\ E_q\ \texttt{[?]},$$

and the target is the entity $E_{2,j}$ from the unique fact whose address matches the query, i.e. $(E_{1,j}, R_j) = (E_q, R_q)$. The model is trained only on the final answer position, with output logits restricted to the 100 entity tokens.

The task is constructed to require matching on the composed address (E_1, R), rather than on either variable alone. Within each prompt, $(E_{1,i}, R_i)$ pairs are unique. In addition, with probability 0.75, we include a distractor fact, sharing the query entity but with a different relation and target. Thus, solving the task requires locating the fact slot whose entity–relation address matches the query and returning its payload entity E_2.

We reverse the query order to R_q, E_q, `[?]` rather than E_q, R_q, `[?]` so it does not repeat the fact prefix order: this removes an induction-head-style continuation shortcut and forces the model to match on the composed entity–relation address.

3. Results

The trained model solves the task perfectly, reaching 100% accuracy on the 20,000-example validation set used during training and 100% accuracy on a separate held-out test set of 8,000 prompts. Unless otherwise stated, all analyses in this section are computed on this held-out test set.

We find that the model solves the task with a staged retrieval circuit (Figure 1). Layer 0 writes two distinct representations to each `[SEP]` position: an address representation for the fact identity (E_1, R), and a payload representation for the target entity E_2. On the query side, Layer 0 also writes the queried address (E_q, R_q) to the final `[?]` position. Layer 1 then compares the query representation at `[?]` to the address representations at previous `[SEP]` positions, selects the matching fact slot, and retrieves the corresponding payload.

Target	Position	Activation site	Accuracy
Fact-side probes after Layer 0 at [SEP] positions			
E_2	[SEP]	Residual stream after Layer 0	100.0%
E_2	[SEP]	Residual stream before Layer 0	1.13%
E_2	[SEP]	Layer 0 head 0 output	0.0%
E_2	[SEP]	Layer 0 head 1 output	100.0%
E_1	[SEP]	Layer 0 head 0 output	100.0%
R	[SEP]	Layer 0 head 0 output	100.0%
(E_1, R)	[SEP]	Residual stream after Layer 0	100.0%
(E_1, R)	[SEP]	Layer 0 head 0 output	100.0%
Query-side probes after Layer 0 at the [?] position			
E_q	[?]	Layer 0 head 0 output	100.0%
R_q	[?]	Layer 0 head 0 output	100.0%

Table 1. Linear probe results. The first head of Layer 0 carries address-side information, while the second carries payload information.

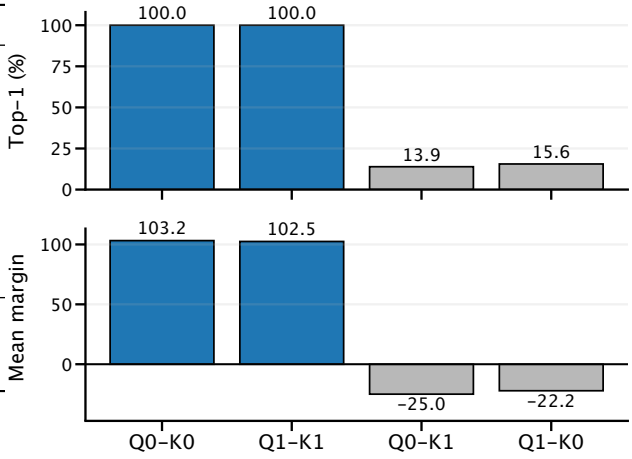

Figure 2. Layer 1 query-key matching. Same-head comparisons select the correct separator with 100% top-1 accuracy and large positive margins, while cross-head comparisons fail. Here $Q0$-$K0$ denotes scores computed from the query and key vectors of Layer 1 head 0, and $Q1$-$K1$ for Layer 1 head 1.

3.1. Layer 0 separates address and payload

Linear probes localize the variables used by the circuit (Table 1). At [SEP] positions after Layer 0, E_2 is perfectly decodable from the residual stream, but not from the residual stream before Layer 0. Head-level probes separate the two roles: Layer 0 head 0 contains the address-side variables E_1, R, and the joint address (E_1, R), while Layer 0 head 1 contains the payload E_2. The query-side variables E_q and R_q are also decodable from the Layer 0 head-0 output at the final [?] position.

3.2. The address decomposes additively

Investigating how the address (E_1, R) is represented, we find that, at the level of (E_1, R) class means, the address is almost exactly explained by an additive decomposition into entity and relation components.

Let $z_i \in \mathbb{R}^{d_{\text{head}}}$ denote the output of Layer 0 head 0 at a [SEP] position for a fact with head entity $E_1 = e$ and relation $R = r$. For each of the 1,000 entity–relation pairs, we compute the class-mean address vector

$$a_{e,r} = \mathbb{E}[z_i \mid E_1 = e, R = r].$$

We then fit the additive model

$$\hat{a}_{e,r} = \mu + u_e + v_r,$$

where μ is a global offset, u_e is an entity component, and v_r is a relation component. This model explains 99.8% of the pair-level variance:

$$\text{FVE} = 1 - \frac{\sum_{e,r} \|a_{e,r} - \hat{a}_{e,r}\|_2^2}{\sum_{e,r} \|a_{e,r} - \bar{a}\|_2^2} = 0.998.$$

The mean cosine similarity between actual and predicted class-mean address vectors is 0.9996. To test that this is genuine factorization rather than a post-hoc fit, we held out 200 of the 1,000 (E_1, R) pairs and fit $\mu + u_e + v_r$ on the remaining 800. The additive model generalized almost perfectly to held-out pairs, achieving FVE 0.998 and mean cosine similarity 0.9995. Applying the same class-level prediction to individual activations still explains 97.7% of activation-level variance, with mean cosine similarity 0.9934.

3.3. Layer 1 retrieves by same-head query-key matching

We next test whether Layer 1 uses the Layer 0 address representations for retrieval. For each Layer 1 head h, we compute the query-key score between the query vector at the final [?] position and the key vector at each preceding [SEP] position:

$$s_h(p) = q^{(h)}_{[?]} \cdot k^{(h)}_p.$$

Here p ranges over separator positions, and $p^\star$ denotes the separator belonging to the correct fact. We define the retrieval margin for head h as

$$m_h = s_h(p^\star) - \max_{p \neq p^\star} s_h(p).$$

Same-head query-key scores select the correct [SEP] in 100% of the 8,000 held-out prompts for both Layer 1 heads (margins +103.2 and +102.5; Figure 2), whereas cross-head scores, query from one head, key from the other, fail at near-chance top-1 with negative margins

3.4. Address interventions redirect retrieval

Finally, we test whether the address representation causally controls which payload is retrieved (Figure 3). We construct mean Layer 0 head-0 address vectors and patch a target-compatible address onto a distractor [SEP] position. This

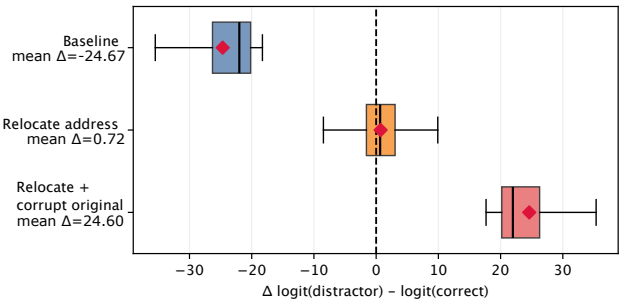

Figure 3. Causal interventions on the address representation. Relocating the target-compatible address to a distractor separator makes the model uncertain; relocating it while corrupting the original target address flips the prediction to the distractor.

intervention alone makes the model uncertain between the original target and the distractor. When we additionally corrupt the original target address, the model decisively flips to the distractor payload. Defining the logit difference as

$$\Delta = \ell_{\text{distractor}} - \ell_{\text{target}},$$

the mean logit difference moves from strongly favoring the original target ($\Delta = -24.67$) to strongly favoring the distractor ($\Delta = +24.60$). The near-symmetric flip is consistent with the address acting as an additive, location-independent key.

3.5. SAE features do not cleanly recover the joint address

The circuit analysis identifies the Layer 0 head-0 output at `[SEP]` positions as the most localized activation site for the relational address, where linear probes decode the joint (E_1, R) perfectly (Table 1). We train BatchTopK SAEs (Bussmann et al., 2024) on two Layer 0 activation sites at `[SEP]` positions: the address-carrying head-0 output and the residual stream after Layer 0. For training, we sweep width $\in \{1024, 2048, 4096\}$ and activation budget $k \in \{4, 8, 16\}$, for three seeds each, to full convergence. Replacing the activations with their SAE reconstruction preserves 100% test accuracy in every run. We therefore ask whether the relational address is recovered as individual SAE features.

For each of the 1,000 (E_1, R) labels, we select the SAE feature with the highest F1 score for detecting that label from feature activation. Figure 4 shows a 2,048-latent SAE with activation budget of 8 trained on the address-carrying head output, where the observed contrast with the probe result is sharpest: the best-matching SAE features for the joint address have high recall but low precision, firing on many occurrences of a given address but also on many other addresses. Across the three seeds, the mean best-feature F1 for joint addresses is only 0.091.

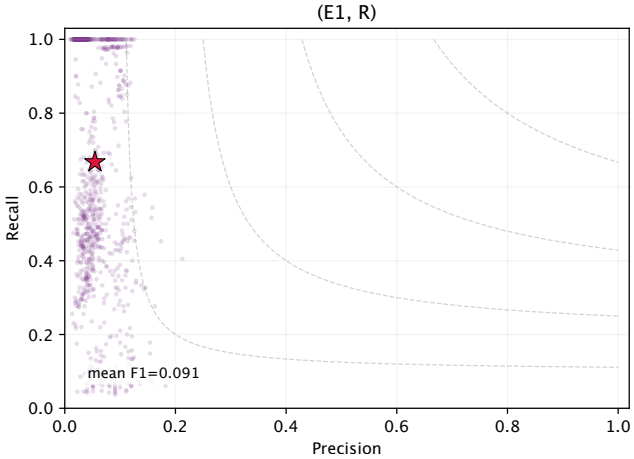

Figure 4. SAE feature recovery of the joint address (E_1, R) from Layer 0 head-0 output at `[SEP]` positions. Each point shows the precision and recall of the best-F1 SAE feature for one entity–relation pair; the red star marks the per-panel mean.

Table 2. Best-feature F1 for the joint address across SAE widths, sparsities, and seeds (BatchTopK, 5M steps; mean ± std over 3 seeds, 8,000-prompt test set).

Activation site	w	k	(E_1, R)
Layer 0 head 0 output	1024	8	.083 ±.004
	2048	4	.081 ±.005
	2048	8	.087 ±.006
	2048	16	.062 ±.002
	4096	8	.081 ±.003
Residual stream after Layer 0	1024	8	.331 ±.081
	2048	4	.267 ±.015
	2048	8	.341 ±.037
	2048	16	.156 ±.007
	4096	8	.390 ±.048

This result extends to all 30 configurations across a variety of widths and activation budgets. We present the results of this in Table 2. Concretely, we find that the mean best-feature F1 is between 0.06–0.09 at the head output and between 0.16–0.39 at the residual stream, against perfect linear decodability from the same activations that we had showcased in Table 1. The gap is consistent across all examined widths and sparsity levels. In Appendix Table 3 we also analyze the recoverability of the individual components of the address, E_1 and R, separately, as well as the payload E_2. Here, we find that these recover more cleanly, and in one case the relation R reaches a near-perfect 1.00 at $k = 4$ and the residual-stream payload E_2 reaches 0.99. Appendix Figures 5–14 show this width-2,048 sweep across $k \in \{4, 8, 16\}$ at both sites. Increasing the activation budget does not clean up the joint address; if anything, the most aggressive sparsity ($k = 16$) degrades both the joint and the marginal recoveries at the head output, indicating that the failure is not a matter of too few active latents.

To analyze whether these poorly-recovered joint latents are genuinely conjunctive, we compare, for each address, the best-F1 (E_1, R) latent against the best-F1 latent for that address's marginal E_1 and marginal R. At the head output with $k = 8$, the best (E_1, R) latent is exactly the best E_1 latent for about 75% of addresses, while the best-R latent almost never coincides: the apparent "joint detectors" are entity detectors, firing on all of an entity's relations with high recall and low (about 0.1) precision. This overlap grows as the dictionary sparsifies, reaching 94% at $k = 4$, with a full per-configuration breakdown in Appendix Table 4. A small minority of addresses do acquire a near-clean dedicated joint latent: taking a best-F1 above 0.8 as the bar, 13–19% clear it at the residual stream, against 0.1% at the head output.

3.6. Groups of latents also miss the address

The previous section ruled out single SAE latents. The address might instead be encoded by a group of latents. We subsequently test the strongest readout of which latents fire: an MLP on the binarized firing pattern, which is the upper bound of any decoder that sees only the on/off code of features. At the head output, where the best single latent reached only F1 0.06–0.09, this group ceiling stays at accuracy 0.51–0.57 across widths ($k=8$), against ≈ 1.0 for a linear probe on the same latents' magnitudes. The on/off pattern discards what the magnitudes carry. At the residual stream, where single-latent recovery was already higher (F1 0.16–0.39), the presence pattern nearly suffices: the same readout reaches 0.95–0.97 ($k=8$) against 0.996 for magnitude. The address is carried in activation magnitudes over a largely shared set of active latents at the localized, additive head output, while the residual stream spreads it across enough latents that the presence pattern is close to sufficient. Full per-width and per-sparsity results are in Table 5.

A related pattern appears for individual entities. Comparing, for each entity, the best latent for its appearances as the address head E_1 against the best latent for its appearances as the payload E_2, the two never coincide: across all residual-stream configurations, no entities share a latent between the two roles. The dictionary represents a single underlying entity as two disjoint, role-specific latent families rather than one reusable feature, a form of feature multiplicity anticipated by Wattenberg & Viégas (2024).

4. Discussion

In a controlled relational retrieval setting, we find that a small attention-only transformer learns an explicit staged mechanism: Layer 0 constructs entity–relation addresses and payload representations, while Layer 1 retrieves by matching the query address against stored fact addresses. The learned address is almost exactly additive and decomposes into an entity component plus a relation component. Together, these results give a concrete example of relational composition that is linearly decodable, causally used, and can be reverse-engineered at the circuit level. The SAE results show a gap between reconstruction and feature recovery. The reconstructions preserve perfect task accuracy, but the joint address does not show up as SAE features. No single latent recovers it, and at the localized head output where the address is constructed, neither does a readout of which latents fire. There the binding is in how strongly latents fire, not which ones.

We reason that because the address is additive, entity and relation directions reconstruct every fact address, and each such direction is reused across all facts that share that entity or relation. With a fixed dictionary width, an SAE minimizing reconstruction error has no incentive to add dedicated pair latents: the $\sim$110 marginal directions already reconstruct all 1,000 addresses, so a latent for a single (E_1, R) pair buys no reconstruction gain. The dictionary it learns is therefore a basis of marginal directions, and the conjunction is missing from it; at the head output it is not even recoverable from a stable group of latents, but spread across the magnitudes of a shared active set. This is the "dark matter" that Wattenberg & Viégas (2024) predict for compositional representations: a combination of features that feature discovery does not surface. They expect such combinations to go missing because they are rare; here the conjunction is common but missing all the same. Reconstruction fidelity and task accuracy therefore do not certify that an SAE has found the variable a circuit actually uses.

The same additive structure produces the multiplicity they also predict. In the residual-stream dictionary, each entity appears under two disjoint sets of latents depending on whether it is the address head or the payload. This is the "echo" Wattenberg & Viégas (2024) anticipate: one entity re-expressed by its role in the relation.

The main limitation is that our evidence comes from a deliberately simplified synthetic setting. This makes the learned mechanism unusually tractable, but does not show whether pretrained language models use analogous address–payload circuits. Future work should test for related mechanisms in fine-tuned or pretrained LLMs, and examine whether findings persists at scale.

Impact Statement

This paper gives a circuit-level account of how a small transformer implements relational binding. The methodological point generalizes beyond this setting where a representation can be linearly decodable while not appearing as individual SAE features, because additive composition and the sparse-feature ontology pick out different kinds of structure. Our evidence is from a synthetic task and a 2-layer model; whether pretrained models bind entities via analogous additive addresses remains open, and is a prerequisite for safety-relevant conclusions from this work.

Acknowledgements

We thank Algoverse AI Research for supporting the early stages of this project through the AI Safety Research Fellowship, including compute support and for bringing the collaboration together. We thank Callum McDougall for feedback on the initial project idea. We also thank the reviewers of the ICML 2026 Workshop on Compositional Learning: Safety, Interpretability, and Agents and the ICML 2026 Mechanistic Interpretability Workshop for comments that improved the paper.

References

Baeumel, T., Gurgurov, D., Al Ghussin, Y., van Genabith, J., and Ostermann, S. Modular arithmetic: Language models solve math digit by digit. In *Proceedings of the 14th International Joint Conference on Natural Language Processing and the 4th Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics*, pp. 1380–1409, 2025.

Bogdan, P. C. and Lindsey, J. Slot machines: How llms keep track of multiple entities. *arXiv preprint arXiv:2604.21139*, 2026.

Bricken, T., Templeton, A., Batson, J., Chen, B., Jermyn, A., Conerly, T., Turner, N., Anil, C., Denison, C., Askell, A., et al. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023.

Bussmann, B., Leask, P., and Nanda, N. Batchtopk sparse autoencoders. *arXiv preprint arXiv:2412.06410*, 2024.

Elhage, N., Hume, T., Olsson, C., Schiefer, N., Henighan, T., Kravec, S., Hatfield-Dodds, Z., Lasenby, R., Drain, D., Chen, C., et al. Toy models of superposition. *arXiv preprint arXiv:2209.10652*, 2022.

Farrell, T., Leask, P., and Moubayed, N. A. Order by scale: Relative-magnitude relational composition in attention-only transformers. In *Socially Responsible and Trustworthy Foundation Models at NeurIPS 2025*, 2025. URL https://openreview.net/forum?id=vWRVzNtk7W.

Feng, J. and Steinhardt, J. How do language models bind entities in context? *arXiv preprint arXiv:2310.17191*, 2023.

Nanda, N., Chan, L., Lieberum, T., Smith, J., and Steinhardt, J. Progress measures for grokking via mechanistic interpretability. *arXiv preprint arXiv:2301.05217*, 2023.

Prakash, N., Rott Shaham, T., Haklay, T., Belinkov, Y., and Bau, D. Fine-tuning enhances existing mechanisms: A case study on entity tracking. In *The Twelfth International Conference on Learning Representations*, 2024.

Wattenberg, M. and Viégas, F. Relational composition in neural networks: A survey and call to action. In *ICML 2024 Workshop on Mechanistic Interpretability*, 2024. URL https://openreview.net/forum?id=zzCEiUIPk9.

A. Appendix

A.1. SAE sweep results

Table 3. Per-component best-feature F1 across SAE widths, sparsities, and seeds (BatchTopK; mean±std over 3 seeds, 8,000-prompt test set). The joint (E_1, R) does not recover; marginals recover better.

Activation site	w	k	E_1	R	E_2	(E_1, R)
Layer 0 head 0 output	1024	8	.51 ±.00	.66 ±.04	.44 ±.01	.083 ±.004
	2048	4	.56 ±.02	1.00 ±.00	.50 ±.00	.081 ±.005
	2048	8	.52 ±.02	.62 ±.03	.42 ±.01	.087 ±.006
	2048	16	.43 ±.03	.61 ±.04	.40 ±.03	.062 ±.002
	4096	8	.51 ±.02	.62 ±.00	.43 ±.02	.081 ±.003
Residual stream after Layer 0	1024	8	.62 ±.13	.55 ±.01	.96 ±.03	.331 ±.081
	2048	4	.85 ±.01	.85 ±.02	.99 ±.00	.267 ±.015
	2048	8	.66 ±.07	.57 ±.01	.94 ±.01	.341 ±.037
	2048	16	.88 ±.03	.36 ±.01	.75 ±.01	.156 ±.007
	4096	8	.61 ±.06	.56 ±.02	.97 ±.01	.390 ±.048

Table 4. Identity of the best-F1 joint-address latent. For each address, we report the fraction of cases where the best (E_1, R) latent is identical to the best marginal E_1 or R latent; role-shared reports the fraction of entities whose best latent is shared between the address-head and payload roles.

Activation site	w	k	$= E_1$	$= R$	role-shared
Layer 0 head 0 output	1024	8	.75 ±.08	.00 ±.00	–
	2048	4	.94 ±.03	.00 ±.00	–
	2048	8	.75 ±.05	.00 ±.00	–
	2048	16	.80 ±.08	.00 ±.00	–
	4096	8	.77 ±.02	.00 ±.00	–
Residual stream after Layer 0	1024	8	.55 ±.16	.01 ±.01	.000 ±.000
	2048	4	.82 ±.02	.00 ±.00	.000 ±.000
	2048	8	.58 ±.07	.02 ±.01	.000 ±.000
	2048	16	.92 ±.02	.00 ±.00	.000 ±.000
	4096	8	.52 ±.09	.01 ±.01	.000 ±.000

Table 5. Recovering the joint address from the SAE latents: linear probe on graded activations vs. the best readout of the binary firing pattern (MLP on binarized latents; bounds any Boolean readout).

Activation site	w	k	graded	MLP-on-bits
Layer 0 head 0 output	1024	8	1.000 ±.000	.568 ±.040
	2048	4	1.000 ±.000	.478 ±.053
	2048	8	1.000 ±.000	.569 ±.075
	2048	16	1.000 ±.000	.810 ±.042
	4096	8	.999 ±.000	.514 ±.006
Residual stream after Layer 0	1024	8	.996 ±.002	.950 ±.050
	2048	4	.999 ±.000	.983 ±.005
	2048	8	.997 ±.002	.960 ±.011
	2048	16	.994 ±.001	.745 ±.054
	4096	8	.996 ±.001	.966 ±.016

A.2. Per-configuration recovery panels

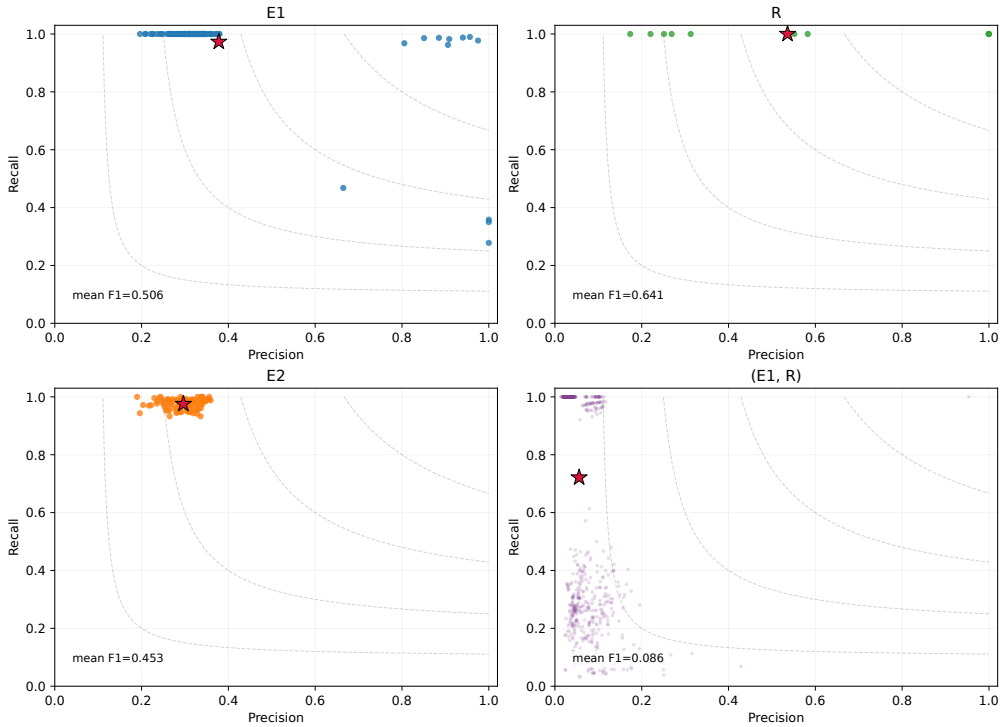

Figure 5. Best-F1 feature recovery, Layer 0 head-0 output, $w = 1024$, $k = 8$. Panels show the marginal labels E_1, R, E_2 and the joint address (E_1, R); each point is the precision and recall of the highest-F1 latent for one label, and the red star marks the per-panel mean.

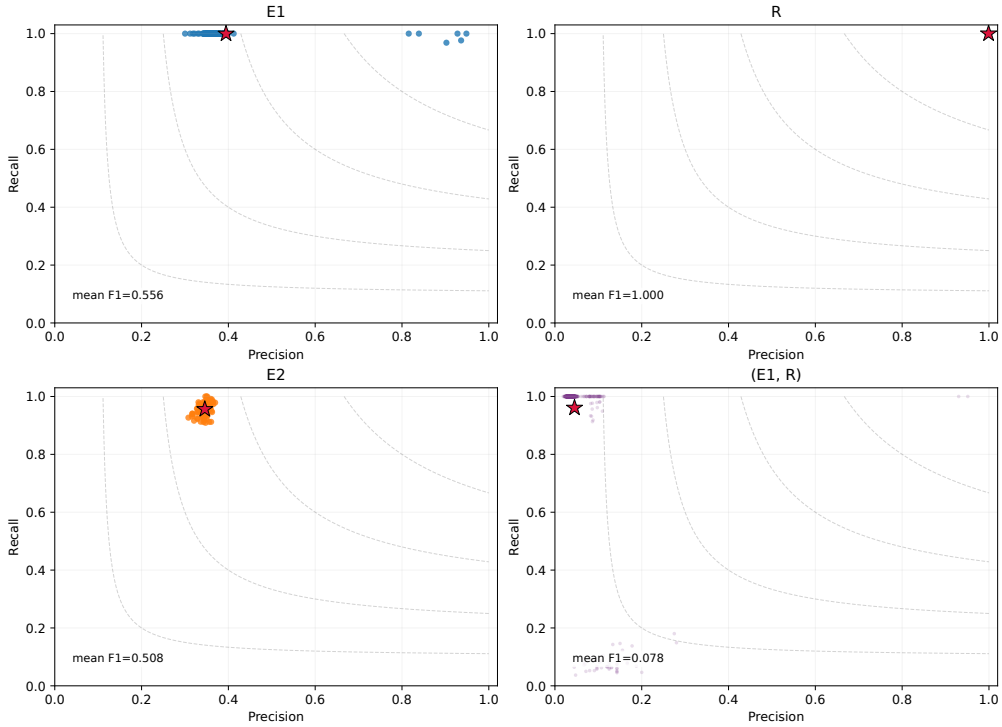

Figure 6. Best-F1 feature recovery, Layer 0 head-0 output, $w = 2048$, $k = 4$. Panels show the marginal labels E_1, R, E_2 and the joint address (E_1, R); each point is the precision and recall of the highest-F1 latent for one label, and the red star marks the per-panel mean.

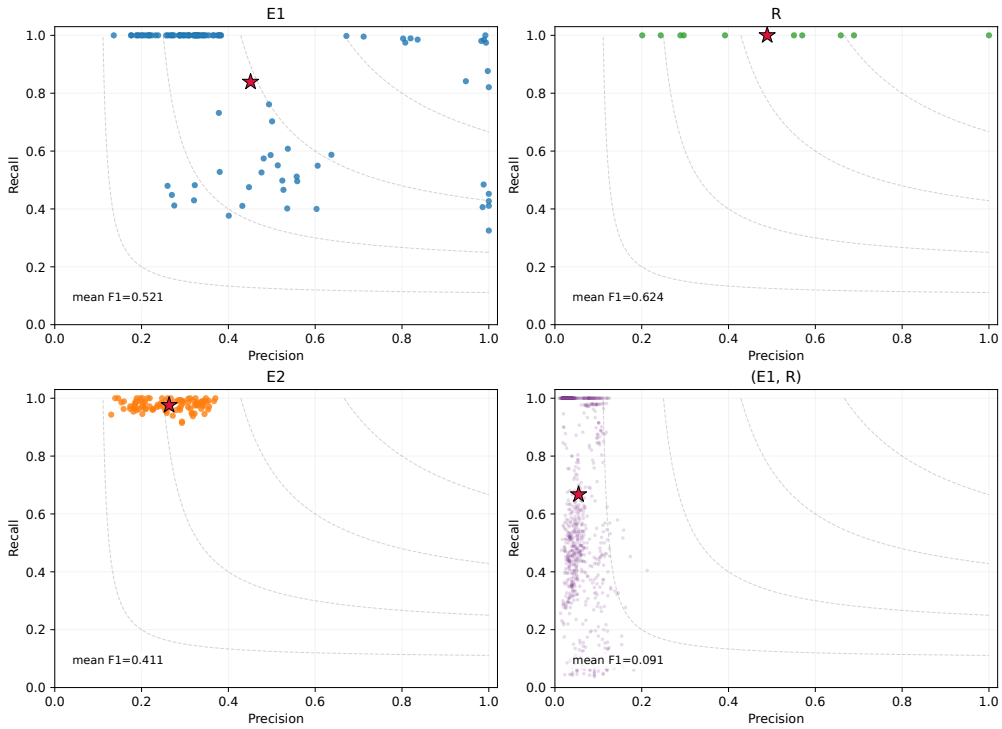

Figure 7. Best-F1 feature recovery, Layer 0 head-0 output, $w = 2048$, $k = 8$ (the main-text configuration, Figure 4). Panels show the marginal labels E_1, R, E_2 and the joint address (E_1, R); each point is the precision and recall of the highest-F1 latent for one label, and the red star marks the per-panel mean.

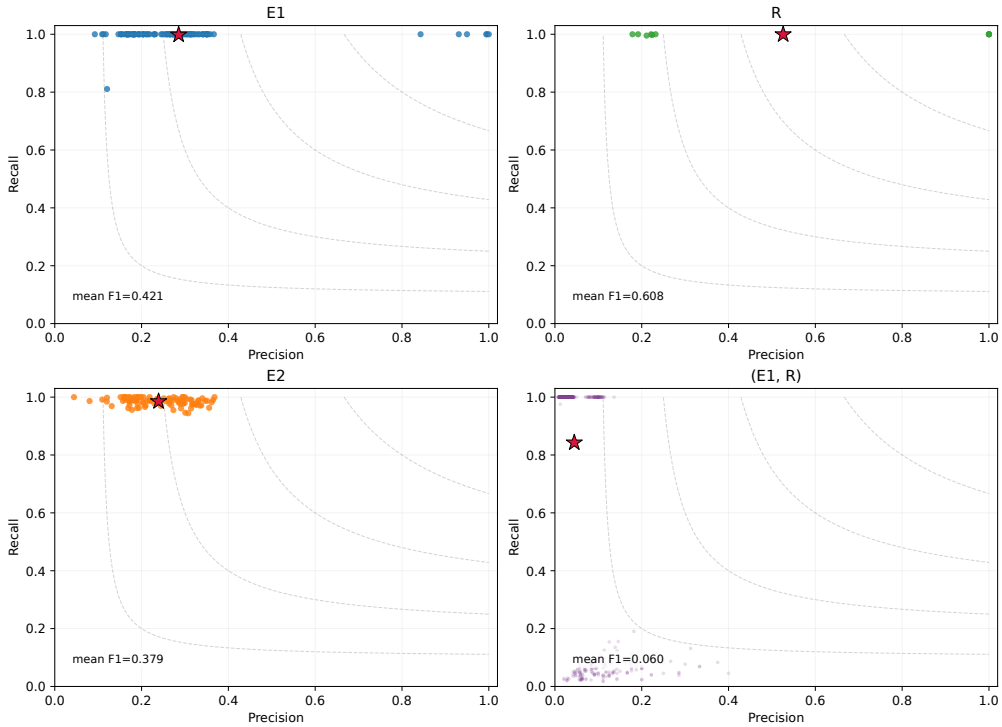

Figure 8. Best-F1 feature recovery, Layer 0 head-0 output, $w = 2048$, $k = 16$. Panels show the marginal labels E_1, R, E_2 and the joint address (E_1, R); each point is the precision and recall of the highest-F1 latent for one label, and the red star marks the per-panel mean.

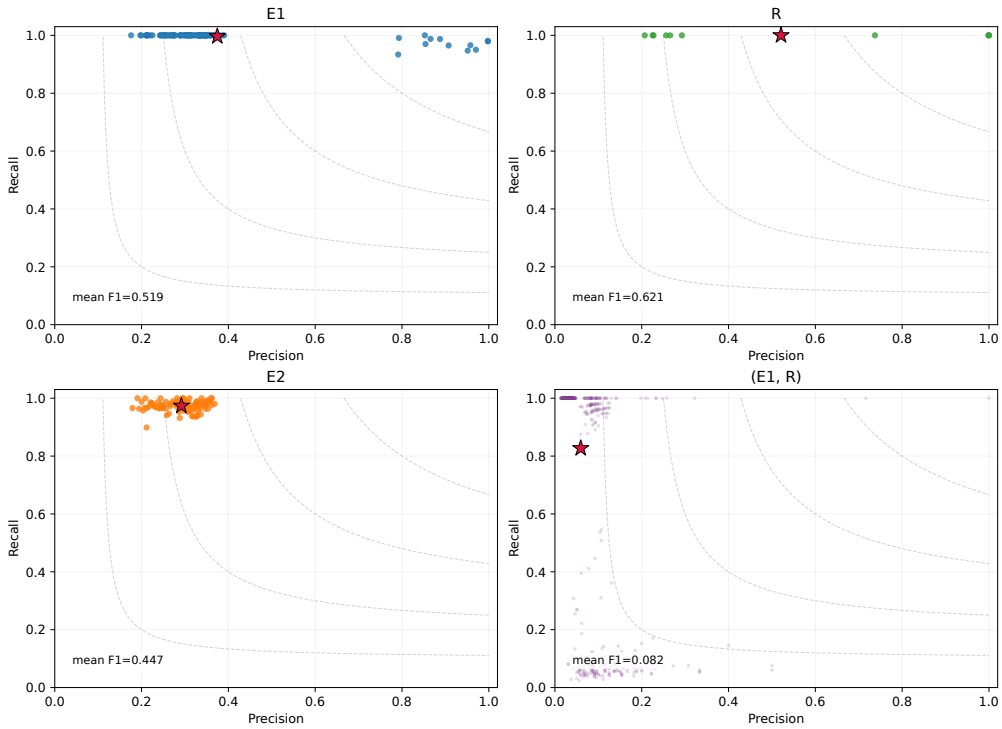

Figure 9. Best-F1 feature recovery, Layer 0 head-0 output, $w = 4096$, $k = 8$. Panels show the marginal labels E_1, R, E_2 and the joint address (E_1, R); each point is the precision and recall of the highest-F1 latent for one label, and the red star marks the per-panel mean.

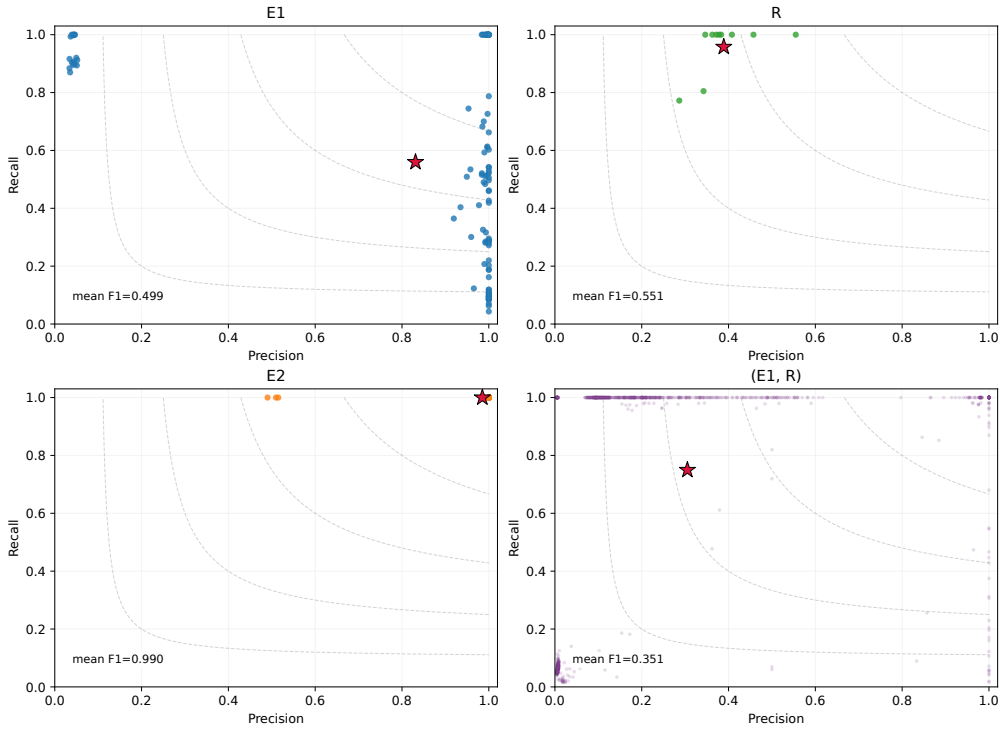

Figure 10. Best-F1 feature recovery, residual stream after Layer 0, $w = 1024$, $k = 8$. Panels show the marginal labels E_1, R, E_2 and the joint address (E_1, R); each point is the precision and recall of the highest-F1 latent for one label, and the red star marks the per-panel mean.

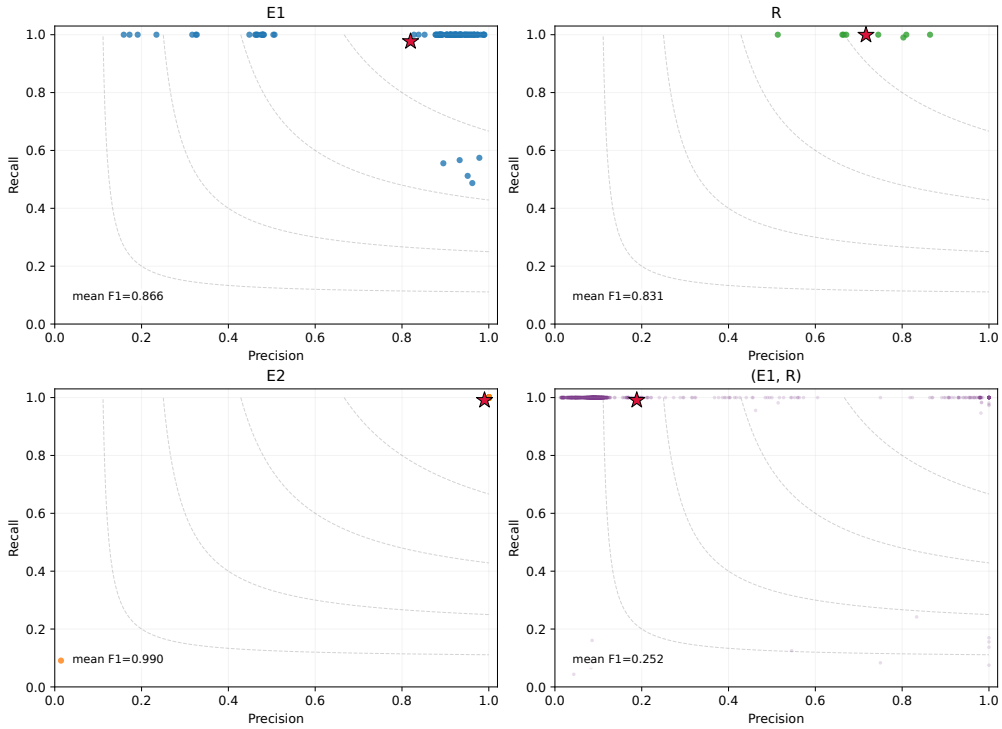

Figure 11. Best-F1 feature recovery, residual stream after Layer 0, $w = 2048$, $k = 4$. Panels show the marginal labels E_1, R, E_2 and the joint address (E_1, R); each point is the precision and recall of the highest-F1 latent for one label, and the red star marks the per-panel mean.

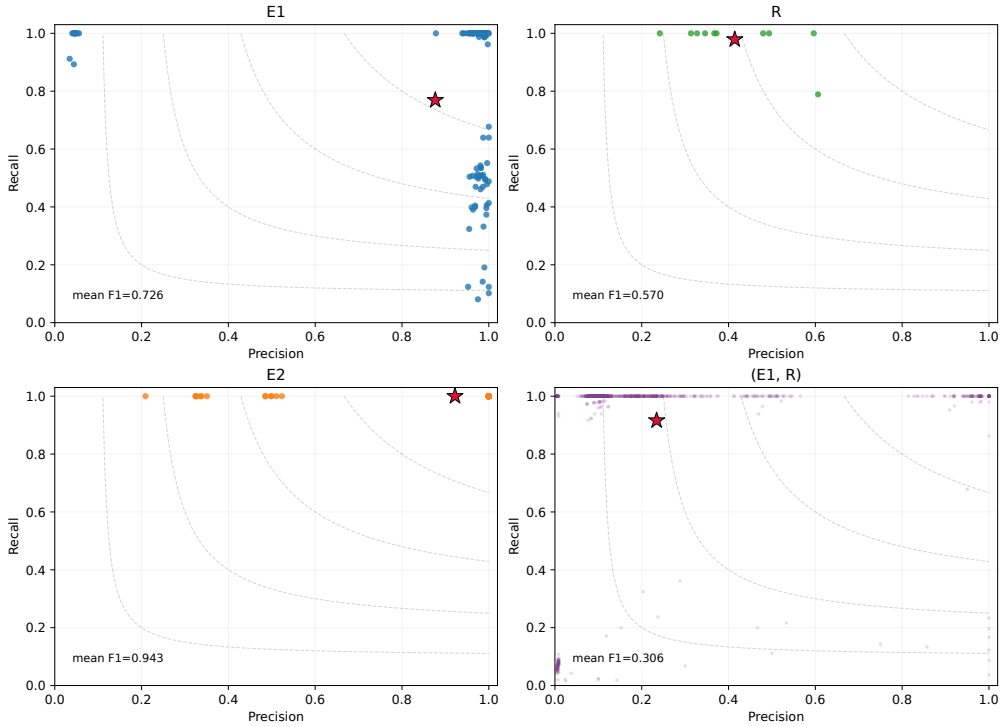

Figure 12. Best-F1 feature recovery, residual stream after Layer 0, $w = 2048$, $k = 8$. Panels show the marginal labels E_1, R, E_2 and the joint address (E_1, R); each point is the precision and recall of the highest-F1 latent for one label, and the red star marks the per-panel mean.

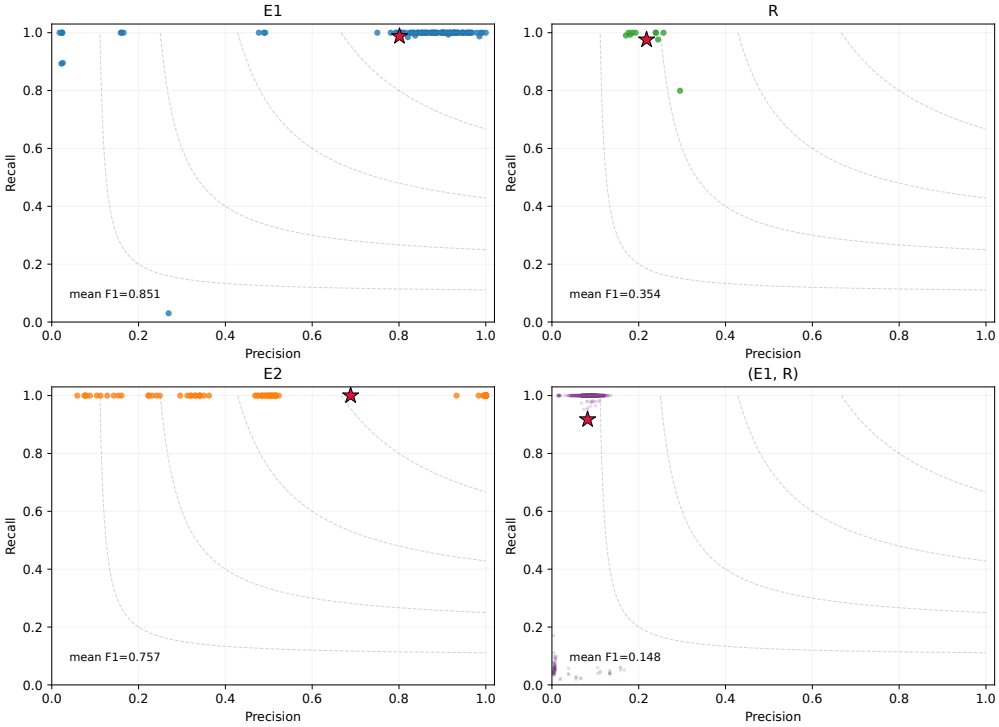

Figure 13. Best-F1 feature recovery, residual stream after Layer 0, $w = 2048$, $k = 16$. Panels show the marginal labels E_1, R, E_2 and the joint address (E_1, R); each point is the precision and recall of the highest-F1 latent for one label, and the red star marks the per-panel mean.

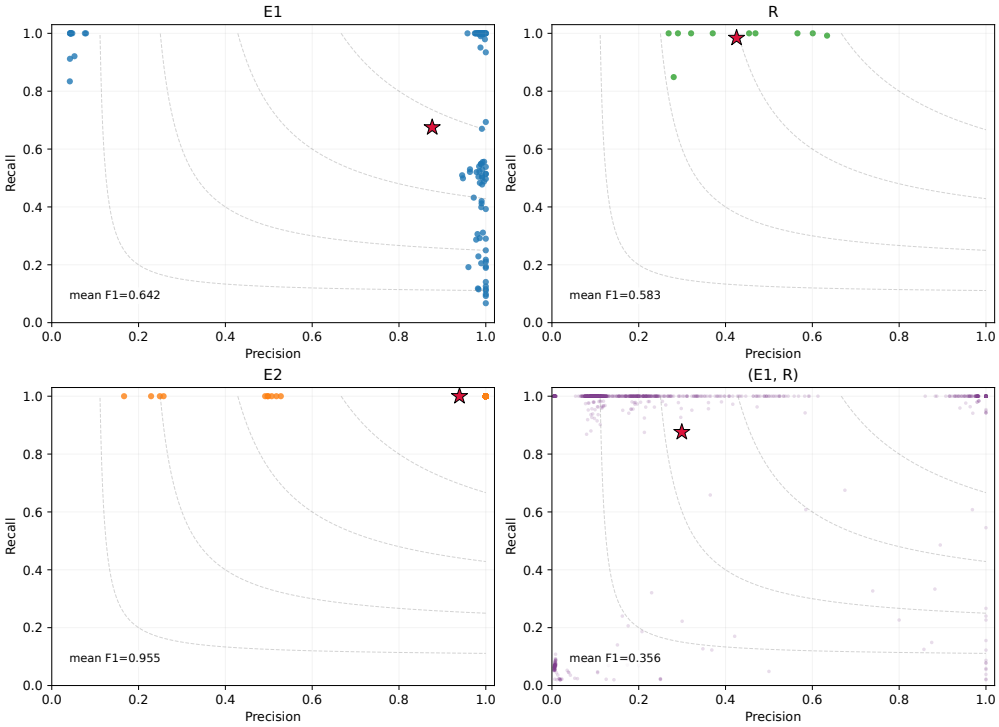

Figure 14. Best-F1 feature recovery, residual stream after Layer 0, $w = 4096$, $k = 8$. Panels show the marginal labels E_1, R, E_2 and the joint address (E_1, R); each point is the precision and recall of the highest-F1 latent for one label, and the red star marks the per-panel mean.