MISPLACED, NOT FORGOTTEN: COMPRESSED CODES MAP UNSEEN ATTRIBUTE COMBINATIONS TO SEEN ONES

Anonymous authors
Paper under double-blind review

ABSTRACT

Models often have to handle combinations of familiar attributes that never appeared together in training, and they often fail on them. Existing work tests whether composition succeeds and which representation geometry supports it. We study where a failing model's predictions go, in models whose heads for several attributes all read one compressed representation, a shared read-out. Such a model maps an unseen combination onto one specific seen combination, a failure we call *capture*. The captured combination is predictable before training from the nearest seen mean image (0.76 against 0.07 chance). The relation is causal. Deleting that combination and retraining moves the prediction far more often than retraining alone, usually onto the next-nearest seen combination, and repeated deletion walks down that ordering. This holds on rendered benchmarks and on real camera photographs. The attributes remain decodable, and a composition operator on the same code recovers held-out combinations at 0.86 where the model scores 0.02, so capture is a failure of placement. Reserving part of the compression budget for each attribute does not reduce capture at matched rate. This holds on a controlled task and on frozen DINOv2, SigLIP and CLIP encoders. Specialising the read-out yields clear paired gains in held-out composition in six of seven settings and a positive but inconclusive one on MPI3D. Decoding every attribute on seen combinations does not guarantee composition. We show where these models send unseen combinations when they fail, and how changing the training support changes that destination.

1 INTRODUCTION

Deployed models are routinely asked about combinations of attributes they never saw together. A household robot may be told to put a familiar object in an unfamiliar receptacle, and a product search may ask for a known style in a new material. On robot goal text of this kind, a shared read-out maps such requests onto seen object–receptacle pairs, usually the nearest in meaning (Table 1). Many such systems compress the input and read several attributes off the result, as do compressed language interfaces such as gist tokens, key-value caches and retrieval embeddings. We write *code* for that compressed representation, as Alemi et al. (2017) do. That such codes handle attribute *combinations* they never saw poorly is established (Montero et al., 2021; Schott et al., 2022; Chaabouni et al., 2020), and a substantial literature asks *whether* composition succeeds (Uselis et al., 2025; Redhardt et al., 2025; Wiedemer et al., 2023) and what geometry it needs (Uselis et al., 2026).

This literature mostly measures whether an unseen combination is recovered. When it is not, accuracy alone cannot say where the prediction *went*, and so cannot tell two failures apart. The code may have *forgotten* the combination, losing information about its attributes. Or it may have merely *misplaced* it, keeping the attributes but putting the input where the read-out finds another combination. The first calls for more bits, the second for a different read-out. A model can decode its seen combinations and keep the attributes, and still fail off support in a direction the support decides.

Our claim is conditional and specific. When several factors are read through a *shared* read-out of a capacity-limited code, the failure is of the second kind. The geometry of the seen support assigns the errors on unseen combinations to one particular seen combination. Specialising the read-out per factor reduces this. Protecting each factor's bits does not. On a frozen language model the causal

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

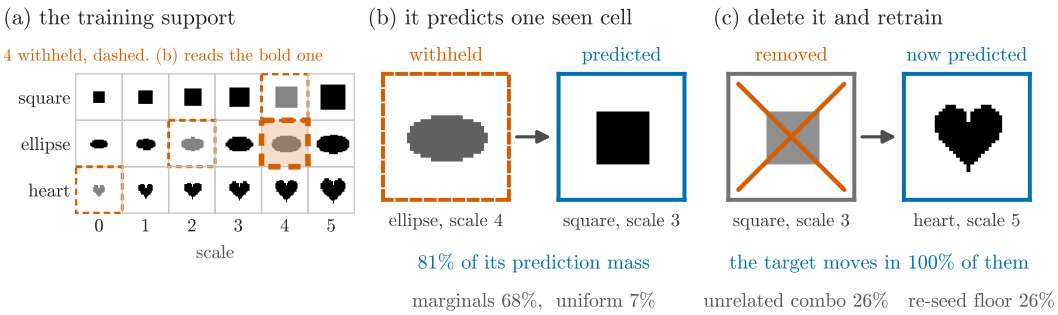

Figure 1: **Capture on dSprites.** Renderings from one plain shared-read-out model (β=0.1, seed 0). **(a)** Four of eighteen combinations are withheld from training (dashed). **(b)** A withheld combination is assigned to one seen combination, fixed by pixel distance before training. **(c)** Deleting it and retraining moves the prediction. Rates under (c) are over 10 seeds and 39 withheld combinations. All 118 cell rows are in Figure 2.

test does *not* hold, and Section 6 reports that null. Everything below concerns trained encoders on factored image, vector and text benchmarks.

Shared codes collapse onto the nearest seen combination. For a held-out cell, a shared read-out puts most of its prediction mass on a single *seen* combination (Figure 1). The factorised head mostly forces the amount (Section 4.1). The *address*, which seen combination receives it, is predictable from the training split and the input alone. The seen cell whose mean image is nearest names the target many times above chance, on dSprites and on MPI3D real camera photographs. On dSprites, frozen ImageNet features, shared factor values and support degree sit at chance. A fixed perceptual confusion would survive deleting the target, and a destination set by the support would not. Delete the target from the training support and retrain, and the prediction relocates in every cell and seed, usually onto the nearest remaining seen cell. Deleting a random non-target, or the non-target nearest the target, relocates it only 0.275 and 0.200 of the time.

We call this *capture*. It moves only *off*-support inputs, so composition degrades before in-support decodability. The input is misplaced, and its attributes are still in the code. On the same dSprites code and heads, a composition operator built from per-factor prototypes recovers held-out combinations at 0.86 where the real held-out input scores 0.02 (Section 4.4).

What reduces capture is read-out specialisation. By *read-out specialisation* we mean that each head's latent subspace carries only its own factor. A hard-wired mask, an alignment objective or a penalty that learns the partition can each supply it. Capture is per-subspace. A block that sees only one factor can only be captured within that factor's values, and capture by the nearest *value of each factor separately* is exactly correct composition. If capture were lost information, a protected per-factor share of the rate should cure it, and it does not. The effect is graded and can be *learned* without the partition. It reproduces on thousands of further models over frozen DINOv2 (Oquab et al., 2024), SigLIP (Zhai et al., 2023) and CLIP (Radford et al., 2021) embeddings (Section 5).

Contributions. **(1) Capture.** Held-out predictions land on one *seen* combination while the attributes stay readable (Section 4). **(2) Predictability.** Which combination captures is predictable before training, from the training split and the input alone (Section 4.2). **(3) Causal relocation.** Deleting the predicted target and retraining moves the prediction far more often than re-seeding or a distance-matched deletion, usually onto the next-nearest remaining combination. Repeated deletion walks down the same ordering. The test needs only a factored label and a partial support, and we report the one interface where it fails (Sections 4.3 and 6). **(4) Specialisation.** Architectures that compose protect each factor's rate, reduce read-out fan-in (how many dimensions each head reads) and disjoin read-out subspaces all at once. At matched rate, protecting rate leaves capture unchanged, because a floored group holds as much information about the other factor as its own. Reducing fan-in alone does not help. Disjoint read-out subspaces reduce capture, gradedly, even when the partition is learned (Section 5, Table 1). These results rest on two measurement preconditions, reading stochastic codes by *sampling* and comparing arms at matched *achieved* rate (Section 3).

2 RELATED WORK

Compositional generalisation. That models generalise poorly to unseen attribute combinations is well documented (Montero et al., 2021; 2022; Schott et al., 2022; Chaabouni et al., 2020), and the proposed causes differ. They include insufficient data diversity (Uselis et al., 2025), scale (Redhardt et al., 2025), or the absence of a compositional support structure (Wiedemer et al., 2023). However, these accounts describe *whether* composition succeeds. We study where a failing model's predictions go, and show that this destination is predictable before training and set by the training support, which separates the candidate causes.

Disentanglement and rate. A long line of work shapes latent structure through rate-side penalties descended from the information bottleneck (Higgins et al., 2017; Kim & Mnih, 2018; Chen et al., 2018; Tishby et al., 2000; Alemi et al., 2017). Locatello et al. (2019) showed unsupervised disentanglement is unidentifiable without inductive bias, and Resnick et al. (2020) tie capacity to compositional behaviour. However, these penalties control how bits are divided among latent groups. We find that protecting each factor's share of the rate leaves capture unchanged, while restricting each read-out to its own subspace reduces it.

The geometry of composable representations. Uselis et al. (2026) establish that compositional generalisation requires linear, orthogonal per-concept representations, and that the degree of such structure in frozen encoders predicts performance on unseen combinations. Related accounts appear in Andreas (2019) and Elhage et al. (2022), and Liang et al. (2025) report the complementary dissociation. On the same models our diagnostic fails to beat their orthogonality score as a *predictor* (Section 5), though every arm that lowers it well below the plain code's value recovers composition. However, these studies are observational. They relate a representation's geometry to how well it composes, while we intervene on the training support and trace where predictions move.

Compressed interfaces and compositional zero-shot learning. Gisting, prompt compression and compressed KV caches put many attributes through one small budget (Mu et al., 2023; Jiang et al., 2023; Ge et al., 2023; Chevalier et al., 2023; Zhang et al., 2023). We test our account there, and the causal test does not hold (Section 6). Compositional zero-shot learning (CZSL) (Misra et al., 2017; Naeem et al., 2021; Nayak et al., 2022; Mancini et al., 2021) studies unseen attribute×object pairs on real photographs (Yu & Grauman, 2014; Isola et al., 2015) and is the closest prior sighting. The closest prior method to our routed arm is Camposampiero et al. (2025), which we compare against directly (Section 5). CZSL protocols exist *because* predictions are biased toward seen pairs, and calibrate against that bias. However, they do not explain which seen combination a prediction lands on. We show that it is predictable before training and causally set by the support, and link the remedy to sparsity-based identifiability (Lachapelle et al., 2022).

3 SETUP AND CONTROLS

Task and rate axis. A factored label $y = (y_1, \ldots, y_F)$ indexes a product of attribute values, with F factors, V_f values for factor f and $[V_f] = \{1, \ldots, V_f\}$. A training support $S \subsetneq \prod_f [V_f]$ omits some combinations, and composition means joint accuracy on the omitted ones. We call each combination a *cell*, *seen* if it is in S and *held out* otherwise. An encoder $q(z|x)$ is trained with per-factor heads and a variational rate penalty $\beta\,\mathrm{KL}(q(z|x) \,\|\, p(z))$. Sweeping β is the compression axis, with data and architecture fixed (Figure 7, with generators, splits and hyperparameters in Appendix B).

Every arm gets the same supervision. An *arm* is one model variant. Every arm and baseline gets the same supervision, the factor count F and per-factor labels on the *seen* support. *None* sees the held-out combinations or is told which latent dimensions carry which factor. Routed blocks are a random disjoint split drawn per seed, which carries no knowledge of the factors, and group sparsity is given no partition at all. *Architectural* overlap is the fraction of dimensions two heads may read in common (0 routing, 1 plain). *Learned* overlap is the read-out weight mass they place on shared dimensions, which can be small when architectural overlap is 1.

Control 1. Read codes by sampling. The rate constraint governs a draw $z \sim q(z|x)$, and reading the posterior mean $\mu(x)$ instead changes the results. Across arms the mean read-out overstates

held-out composition by up to 0.48. It stays near ceiling long after the sampled read-out has fallen, which is where capture lives, and the gap depends on rate. The factorial and Table 4 report both read-outs, and sampling lowers every level while changing no ordering. Section 4 uses the *sampled* read-out throughout, from a pre-registered re-run in which its three findings (concentration, prediction before training, relocation on deletion) survive a single draw. The posterior-mean values are in Appendix E for comparison. By *pre-registered* we mean thresholds and a sha256 of the analysis script fixed before any outcome was measured, with the registrations shipped in the software release.

Control 2. Match achieved rate. Two arms at the same β land at different achieved rates, so matching β confounds the intervention with the budget it reached. We sweep β per arm and read all arms at common achieved rates by per-seed interpolation along each arm's own frontier, saying so where an arm cannot reach one. A matched-KL InfoNCE check confirms the factorial arms share the $\mathrm{KL} \to I(Z; X)$ conversion to within 0.30 nats. The interpolation therefore compares like information contents.

The metric is sound at both ends. Absolute composition here is low. The *ceiling* is attainable, since a specialised arm reaches 0.65 sampled on dSprites (Matthey et al., 2017). How high depends on *which* combinations a seed withholds (Appendix D). The *floor* carries signal too, since a shuffled-label control collapses to zero while arms at chance still fit the seen support.

4 Capture: where off-support predictions go

4.1 Concentration comes from the architecture, the address from the support

Ask what the model predicts for a held-out cell when held-out composition is at zero. On dSprites the answer is a single *seen* combination (Figure 1 for one cell, Figure 2 in aggregate). It takes 0.830 of the mass, and almost all error mass lands on combinations that *were* in training. Targets differ across 98.3% of held cells, so no single combination attracts everything. Four further image settings concentrate on a *seen* combination in 100% of 231 window rows, and the concentration appears in the code geometry even without the heads (Appendix E). The *original* 3DShapes hue grid has no capture regime, and coarsening it produces one.

The *amount* of concentration follows from the architecture. With per-factor heads the prediction factorises, $p(y \mid z) = \prod_f p(y_f \mid z)$ at every code, so the joint mass concentrates as soon as each head is confident. The product of the two marginals already reaches 0.818 of the 0.830 observed, and at LM scale an uncompressed control errs just as concentratedly (Appendix I). What this paper claims concerns the *address*, which seen combination receives the mass. Section 4.2 shows that it is predictable before training, and Section 4.3 that it moves when the support does.

Units. Each dSprites run holds out 4 of 18 cells, so R rates and K seeds give $4RK$ *cell rows*. The counts below (118, 40, 80) differ only in how many rates and seeds an analysis spans. Rows from one model are not independent, so intervals are *seed-clustered* bootstraps unless stated. The posterior mean gives 0.886 in place of 0.830 (Appendix E).

4.2 The target is predictable before training

If capture is real, the target should be predictable from the training split and the input, with no access to the trained model's errors. It is. On dSprites the seen cell whose mean **image** is nearest in pixel space, computed from the training split alone, names the captured target at $\mathbf{0.761}$ $[0.58, 0.91]$ against a 0.071 chance rate. Three support-blind controls, frozen ResNet18 features, sharing a factor value, and the highest-degree neighbour, all sit at chance. The rule is therefore not a generic similarity effect. The nearest seen *code* reaches 0.950 but reads the trained model and is partly tautological (Appendix E).

Predictability replicates on two further image benchmarks, including real photographs. The same pixel-space predictor, unchanged, identifies the captured target on MPI3D *real camera images* and on Colored-MNIST at $21\times$ and $8.8\times$ their chance rates (Appendix E). *Which* controls fail varies by dataset. On Colored-MNIST the pixel-nearest cell always shares a factor value, so support degree ties the predictor, and on MPI3D frozen ImageNet features also predict.

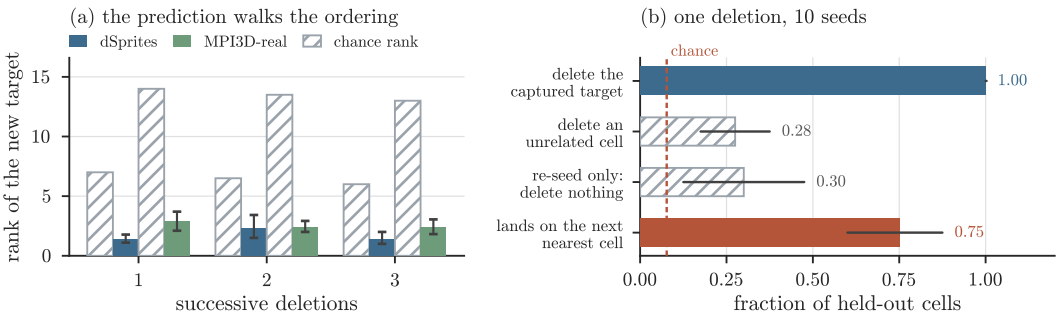

Figure 2: **Where the prediction goes after deletion. (a)** Rank of the new target in the pixel ordering fixed before training, observed (solid) against each grid's chance rank (hatched), over repeated deletions. **(b)** A single deletion on dSprites (β=0.01, 10 seeds) with its controls, shown in (b) alone because their rank is defined only on cells that moved. Whiskers are seed-bootstrap 95% intervals.

What decides the target is a selection rule. Why does a shared read-out return one particular *seen* combination, and when? The factorised head fixes the amount of concentration (Section 4.1). What supplies the *address* is a selection rule.

Proposition 1 (Nearest-support selection)**.** *Let the codes of seen cells $s \in S$ form clusters with means m_s and common isotropic covariance $\sigma^2 I$, and let head f be the Bayes read-out of y_f under that seen-cell mixture, $p(y_f{=}i \mid z) = \sum_{s\in S:\, s_f=i} r_s(z)$ with $r_s(z) \propto \exp(-\|z - m_s\|^2/2\sigma^2)$. Let $s_1(z)$ be the seen cell nearest z and $\Delta(z)$ the gap between the nearest and second-nearest squared distances.* ***(i)*** *If $\Delta(z) > 2\sigma^2 \ln(|S| - 1)$ then $r_{s_1}(z) > \frac{1}{2}$ and every head's argmax is the corresponding value of $s_1(z)$, so the modal joint prediction is the seen cell $s_1(z)$, whether or not it is correct.* ***(ii)*** *Refitting the read-out after deleting s_1 from S gives $s_2(z)$ under the same condition on $S \setminus \{s_1\}$, and so on. Successive deletions walk the distance ordering of S from z.* ***(iii)*** *If head f instead reads only a block z_f whose seen-cell means depend on s_f alone, it minimises $\|z_f - m_i^{(f)}\|^2 - 2\sigma^2 \ln \deg_S(i)$, where $\deg_S(i)$ is the number of seen cells carrying value i, so it selects the nearest* value *of f when those counts are equal and otherwise trades distance against them. The modal joint prediction need not be a seen cell, and the cross-factor derivative $\partial \log p(y_f \mid z)/\partial z_{f'}$ vanishes, whereas under a shared read-out it is generically non-zero.*

The proposition takes the code as given. What it needs from the encoder is that the trained code keep the pixel-space ordering of seen cells, and that holds only in part. On the dSprites capture rows it holds in about two thirds of cells. On geometry fixed before training it reaches 0.341 against a registered 0.5 bar (Appendix E.2). The encoder builds its own metric. That is the open problem we leave, and a registered before-training predictor for it met its bar on one benchmark of three (Appendix N). In the code itself the nearest seen cell names the target at 0.950. Section 4.3 tests part (ii) with a stronger intervention than it states, retraining the encoder where the proposition refits only the read-out. The proof is in Appendix N.1, with a distance–success correlation that the dominance condition does *not* explain.

The proposition predicts the selection and tracks the mass. Every quantity in r_{s_1} is measurable (seen-cell mean codes as prototypes, pooled within-cell variance as σ^2), so part (i) is tested with nothing fitted, on heads never trained toward it. Over 80 held-out cells the formula names the captured cell in 93.8% of them, and predicted (r_{s_1}) and observed modal mass correlate at $r = 0.830$ $[0.762, 0.884]$. The product of the per-factor responsibility sums, which is what factorised heads output under the proposition, tracks the observed mass equally well (0.817). Both overshoot the heads' confidence, by 0.100 and 0.071, so the formula gets the selected cell right and overstates its mass (Appendix N.1). Figure 11 maps where it holds. It survives the violated isotropy assumption, fails where in-support accuracy collapses, and is untested at the dominance boundary, which no cell reaches (pre-registered, Appendix N.1).

4.3 Deleting the target relocates the prediction and it re-captures

A fixed similarity metric predicts capture, but it would equally predict a static confusion that the encoder shows whatever it was trained on. To separate the two we intervene on the support. For

each held-out cell we delete its captured target from training, retrain from scratch, and ask where the prediction goes. It moves in every cell of every seed at both rates, and it is neither repaired nor scattered. It usually lands on the combination that is now the pixel-nearest *remaining* seen cell, far above chance (Table 8). The move is specific to the cell the read-out selected. Deleting an equally sized seen combination that is nobody's target, or the non-target cell *nearest to the target*, moves the prediction far less often (0.275 and 0.200, the second pre-registered, Appendix E). Off the support the network behaves like a nearest-prototype classifier *conditioned on the current support*, which a support-blind confusion cannot mimic. Table 13 lists every rival account we could construct, each with the test that decides it.

The intervention separates capture from two independent per-factor biases. Under a two-marginal account each head prefers a factor *value*, so deleting a single *combination* whose two values both remain elsewhere in the support should move nothing. Restricting to exactly those deletions, 37 of 40 cells at β=0.01, the prediction still relocates in every one. Paired on the same cells, it moves where re-seeding or the matched control leave it in place, never the reverse, at both rates (Table 13). The output nearly factorises into per-factor marginals, yet it responds to deleting one combination.

Why decodability outlasts composition. Capture moves only off-support inputs, so a probe on the seen support reads it unchanged (seen-conjunction probe 0.95 against held-out composition 0.60). That is why decodability and composition come apart over a band of rates (Appendix E).

Deletion outpaces re-seeding in every cell. Relocation after a deletion is evidence only if retraining does not relocate the prediction anyway. So we add a *re-seed* arm, which deletes nothing and redraws the training seed, and run all three arms under one protocol on four benchmarks (Table 8). On dSprites re-seeding moves the prediction in 0.300 of cells, slightly more than deleting an equally close non-target, because it varies the training randomness where the deletion arms vary the support. Deleting the target exceeds both in every cell, by $+0.700$. We call it *re-capture* when the relocated prediction lands on the nearest remaining seen combination. Even against a null that credits a shared factor value, re-capture runs at 2.7 to $3.7\times$ chance (Table 13). On coarse-hue 3DShapes re-seeding alone moves 0.950 of cells, leaving nothing to attribute to the support, which the original protocol could not have detected.

The relocation is a cascade, on rendered and on real images. Proposition 1(ii) says deleting the new target too should hand the prediction to the third-nearest cell, and so on. Iterating the deletion to depth three (pre-registered, 5 seeds), the prediction relocates at every step and lands on the pixel-nearest *remaining* seen cell, ranked from the image bank before training (Figure 2a). On dSprites it lands there on 0.704 of chain steps against 0.083 chance. On real camera photographs (MPI3D-real) it lands there on 0.457 against 0.038 chance. Random deletions and re-seeding relocate it at most half as often (Table 12). On MPI3D-real thirteen of its 14 pre-registered tests pass. The one that fails is this pooled top-one rate, which is below the 0.5 bar set on dSprites.

4.4 THE FACTORS ARE STILL IN THE CODE

The last alternative is that the code has simply lost the factors. Two tests show it keeps them. The first asks the *encoder*. On MPI3D-real, conditioned on the held-out cells themselves, the code keeps 2.06 of the 2.94 nats about their factors, enough under Fano for joint held-out accuracy of 0.80 where the heads deliver 0.03. The second asks the *read-out* and needs no information estimate. Through the *same* trained heads of the *same* plain dSprites code, a composition operator built from per-factor prototypes recovers held-out combinations at 0.858, while a real held-out image scores 0.020 (both in Appendix C). The factors are present and readable. What the plain code gets wrong is *where* the encoder places an unseen input.

A formal certificate (Fano with data processing, Proposition 2) bears only on whether the information is *necessary*, and only in part. Certified factor information falls 0.235 nats short of its ceiling while composition falls by 0.35, too small a deficit to *force* the error, and the pre-registered criterion (0.051 nats) is met on neither image dataset. The claim therefore rests on the two tests above, which show that the input keeps its factors and is only misplaced.

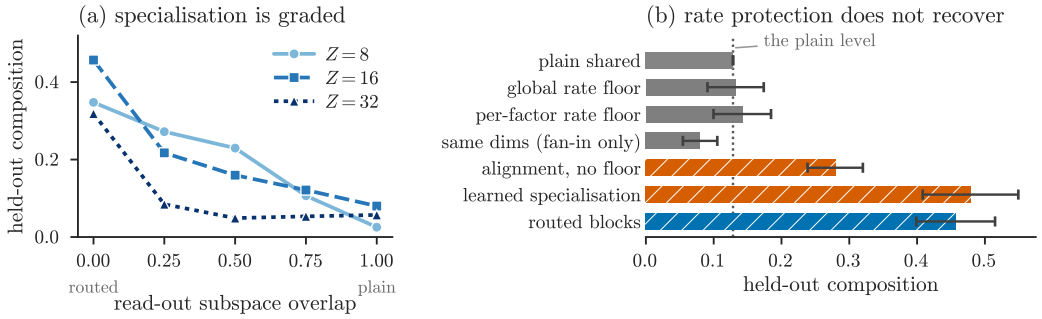

Figure 3: **Read-out specialisation reduces capture. Rate protection does not.** Synthetic 6×6, 10 seeds, sampled read-out, matched achieved rate. **(a)** Held-out composition against the overlap of the two heads' read-out subspaces at fixed fan-in, for three latent widths D (the D=32 tail is non-monotone). **(b)** Held-out composition per arm, with 95% paired intervals. All 15 arms are in Table 16.

5 Mechanism: Read-out specialisation beats rate allocation

What in a factored architecture reduces capture?

5.1 The obvious fix fails, and the instrument says why

If a shared code captures because its factors compete for one budget, the natural remedy is to give each factor a protected share of the rate. Flooring each latent group's rate does not recover composition ($+0.014$, interval spanning zero), and it is indistinguishable from a *global* total-rate floor at all three latent widths and on every frozen encoder (1148 floored models). Added to routing it *hurts*, by -0.076 $[-0.108, -0.045]$. Measuring the per-group *information* about each factor shows why. A KL-floored group holds 0.675 nats about its own factor and 0.661 about the other, statistically the plain code. A rate floor sets how many bits a group spends and leaves open which factor they describe.

No cited work proposes per-factor floors as the mechanism. What does restore composition at matched achieved rate is one of three interventions, routed blocks, independent per-factor bottlenecks, and independent bottlenecks over a shared trunk (Appendix F). All three change three things at once. They protect a per-factor rate, halve the read-out fan-in, and make the read-out subspaces disjoint. The rest of this section separates the three, with arms that isolate each and a sweep of the heads' subspace *overlap* at fixed fan-in (Figure 3, all 15 arms in Appendix G).

An instrument that *does* allocate information works by specialising. So we built one. It penalises the shortfall of $I(Z_g; y_f)$ below a target and leaves the read-out unrestricted. It allocates information as intended, 1.180 nats about its own factor against 0.112 about the other, and recovers about a third of the routing gap (Appendix G). It does so by specialising. We measure specialisation with the *crossed probe*, the accuracy of a linear read-out of one latent group alone on the *other* factor. It is low when each group carries only its own factor. Within the *crux band* (1.5–5.0 nats, the range where the arms separate), the information floor lowers the probe from the KL-floored arm's 0.884 to 0.430, near routing's 0.333, and composes no better than that probe value predicts (Appendix G). Giving a factor's information its own subspace is itself a way of specialising it.

The probe is a manipulation target first and a predictor second. Figure 4 places every arm of the factorial on both axes. The four arms that recover the most all specialise the read-out. Arms that floor a group's rate or share read-out dimensions keep the probe near or above the plain code's value, and recover at most a third of the gap. Reduced fan-in is ruled out. The arm whose two heads read the *same* random half falls below the plain code, with 99.9% of its rate inside the dimensions they read. Group sparsity is the one exception on the probe. It reaches routing-level composition while its probe stays at the plain code's value (0.865 vs. 0.861), because it *discovers* its own partition, which the probe's *pre-specified* groups cannot see. As a *predictor* the probe adds little. A pre-registered head-to-head against the linear-orthogonality score of Uselis et al. (2026) went against us. Over 2156 models and six cells it cleared its bar in 2 where 4 were required.

Composition falls as read-out subspaces overlap. Sweeping overlap at fixed fan-in, composition falls with slope -0.340 $[-0.382, -0.298]$ sampled (-0.586 mean). Sharing 2 of 16 latent dimensions already removes 73% of routing's benefit, replicating at widths 8 and 32.

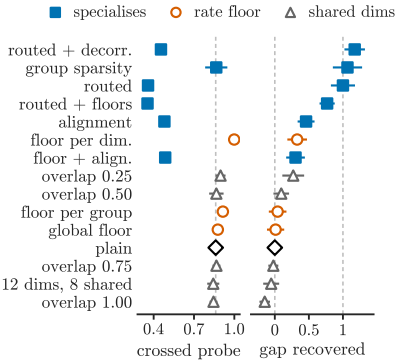

Figure 4: Every arm of the factorial, sorted by gap recovered, with 95% paired intervals over 10 seeds (crux band, sampled read-out). Dashed lines mark the plain code and routing.

Specialisation can be installed without a mask, and can be learned. An alignment objective adds auxiliary heads that each read one half of the code, with no rate floors, while the main heads still read all of it. It recovers 46% of the routing gap, so arranging the code by factor helps, and restricting what each head reads helps more. A group-sparsity penalty that must *discover* where to specialise recovers all of it, converging to head supports whose *learned* overlap is near zero under an unrestricted mask, robust to the number of latent groups it is handed. In a separate method comparison (Table 15) it matches routing on frozen DINOv2 embeddings and recovers 92% of the gap on UT-Zappos50K. An attribute-invariance objective, itself a specialiser, ties it on Zappos and beats both it and routing on 3DShapes, as the mechanism predicts. Every objective that specialises the read-out clears every objective that shapes the rate, on both datasets, and the β-VAE family sits at or below the plain code at its best swept hyperparameter.

The whole ordering replicates on frozen foundation encoders. We re-ran the identical factorial on cached embeddings of three frozen encoders and a second dataset (MPI3D, pre-registered), 4922 further models (Table 18). Every arm-level ordering reproduces (Table 1), with floors near zero effect, fan-in alone at or near the plain level, every per-seed overlap slope negative, and learned group sparsity recovering 72–98% of the gap at a learned overlap near zero. The crossed probe remains the dominant across-arm predictor (-0.70 to -0.80 on 3DShapes, -0.61 on MPI3D), while rate balance's sign is unstable (three bounded exceptions in Appendix H). Entered jointly, model by model inside the crux band, overlap and the crossed probe each carry an independent negative effect on composition at fixed rate, with intervals clear of zero, while rate keeps a positive one. Bits are necessary, and at matched bits the read-out geometry decides (Appendix G). Forcing the encoder to be additive in factor prototypes closes the additivity gap on the seen support while the off-support gap barely moves (Appendix J).

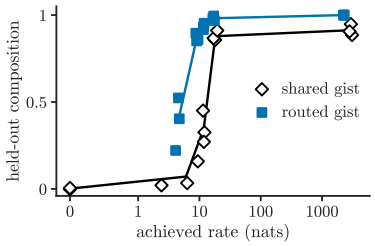

Figure 5: **Specialisation in a frozen LM.** Gist codes over Qwen2.5-1.5B, each seed at its own achieved rate (Appendix L).

6 Generality and Boundaries

All three findings hold on real camera photographs. On MPI3D-real, photographs of a physical robot platform, all three findings hold at one pre-registered rate, all four pre-registered tests pass, and so does the depth-three cascade at 13 of 14 (Table 8, Appendix E). Its re-seed floor, 0.075 against dSprites' 0.300, is the lowest we measure, so the causal contrast is cleanest on the least synthetic data. Table 1 (bottom) collects every setting beyond rendered grids (Appendix H).

A positive mean gain in all seven settings, clear in six. Across seven settings, from vectors to robot goal text, a factor-specialised code beats a plain shared token at matched rate on average, with paired intervals clear of zero in six (Table 4). On MPI3D it is positive in 7 of 10 seeds but inconclusive, and clear on frozen DINOv2 features of the same images (Table 1). It also holds in probes over frozen LM hidden states in two model families, Qwen2.5 and SmolLM2, up to 7.6B parameters (Figure 5, Appendix L), and on six frozen foundation encoders, where tightening the budget from 12 to 4 nats costs a plain bottleneck three times what it costs a specialised one (Table 20).

Table 1: **Read-out arms on six settings (top) and capture beyond rendered grids (bottom).** Top, sampled held-out composition at matched achieved rate, crux-band mean over 10 seeds (6 on MPI3D) with 95% intervals. Bold marks the best arm per column and any arm not significantly below it (paired over seeds). Bottom, the target named before training against chance, and the fraction of predictions moved by deleting it against re-seeding alone.

		3DShapes, frozen encoder (hue×scale)				MPI3D
read-out arm	synthetic	DINOv2	SigLIP	CLIP	CLIP†	DINOv2
chance	1/36	1/80	1/80	1/80	1/40	1/36
plain shared	0.13±0.02	0.05±0.01	0.03±0.00	0.03±0.00	0.09±0.02	0.08±0.02
global rate floor	0.13±0.04	0.06±0.01	0.04±0.01	0.04±0.00	0.09±0.02	0.09±0.02
per-group rate floor	0.14±0.04	0.06±0.01	0.04±0.01	0.04±0.01	0.10±0.02	0.09±0.02
per-dimension rate floor	0.24±0.04	0.06±0.01	0.04±0.01	0.04±0.00	0.11±0.03	0.10±0.02
same dims (fan-in only)	0.08±0.02	0.05±0.01	0.04±0.01	0.03±0.01	0.05±0.02	0.07±0.02
overlap 0.50	0.16±0.04	0.10±0.02	0.08±0.02	0.08±0.02	0.08±0.03	0.10±0.02
routed blocks	**0.46**±0.05	**0.27**±0.02	**0.26**±0.01	**0.24**±0.01	**0.30**±0.04	**0.30**±0.04
group sparsity, learned	**0.48**±0.08	0.25±0.04	0.20±0.03	0.20±0.03	0.27±0.04	**0.29**±0.06

beyond rendered grids	MPI3D-real	frozen encoders[a]	ALFRED goal text	frozen LM[b]
predicted / chance	0.750 / 0.036	0.66–0.68 / <0.01	0.602 / 0.018	0.214 / 0.028
deleted / re-seed	1.00 / 0.075	1.00 / 0.11–0.33	0.911 / 0.327	0.958 / 0.875, **fails**

†hue×shape. MPI3D is colour×shape. [a]DINOv2, SigLIP and CLIP, prediction on CLIP and SigLIP, deletion exploratory. [b]Qwen2.5, where re-seeding alone moves the prediction.

Decodability does not certify composition, on real photographs. We measure this on 16 cells of real photographs (CelebA, UT-Zappos50K and Fashion-Product-Images, over frozen CLIP and DINOv2). Joint linear-probe accuracy and held-out composition come apart ($r = 0.50$), and in the ten cells where *both* attributes decode at joint probe ≥ 0.5, composition never exceeds 0.73 of the cell's chance rate. The best-decoded cell of all, at probe 0.82, composes at 0.18 against a chance of 0.25. An audit that probes each attribute separately would pass every one of those embeddings. Two caveats apply. The CelebA cells are 2×2 with severe class imbalance, so the held-out cell is usually the rare one, and the routing advantage there is relative, bounded by whether the frozen encoder carries the attribute at all (Appendix A).

Capture does not replicate on a frozen LM. Splicing one gist-compressed instruction's operation block onto another's format block and asking a frozen Qwen2.5 to generate yields the correct novel combination 38.2% of the time at 1.5B, an *existence* result only, since the arms are not rate-matched. The code stays compositional and the failure sits in the decoder, so routing preserves factor *identifiability* without giving *controllability* (Appendices I, M). The boundary is the frozen decoder. On ALFRED goal text, with a *trained* encoder, capture holds (Table 1, Appendix B).

7 Conclusion and limitations

On a combination it never saw, a shared read-out sends its prediction to one specific *seen* combination, usually the most similar. Changing the training support changes that destination, and deleting it moves the prediction to the next. The failure lies in *where* the code places an unseen input, which still carries its factors. Specialising the read-out reduces it. Protecting any factor's share of the rate does not. Two precautions follow. First, measure composition on held-out combinations, and where a failed prediction goes. In-support decodability is exactly what capture leaves intact, and the posterior mean overstates held-out composition by up to 0.48. Second, specialise the read-out, since a rate floor does not restore composition and every specialising objective does. The claim is conditional. We establish capture for shared read-outs of trained encoders on two- and three-factor grids. All three findings hold on real camera photographs. There the causal test runs at a single rate, and the cascade's top-one rate (0.457) sits below a bar calibrated on a smaller grid. Proposition 1 explains which seen combination is selected given the code, and leaves open why the encoder builds that geometry. On a frozen language model the failure moves into the decoder, the boundary of the claim (Appendix A).

AI USE STATEMENT

We used generative AI tools to implement methods, design experiments, create code and figures, and draft and edit the paper. We did not use them to generate data or formulate mathematical claims; the hypotheses behind Propositions 2 and 3 came from the authors' observation and intuition. The proof was helped with an AI assistant, with each step checked by the authors. Pre-registrations fixed experimental thresholds before outcomes were measured, and a released audit re-checks every reported number. We reviewed all AI-assisted work and take full responsibility for the content of this paper.

REPRODUCIBILITY STATEMENT

Code, results and all pre-registrations are available at `https://anonymous.4open.science/r/capture-code-6CF2`. The proof of Proposition 1 is in Appendix N.1 and is also checked numerically, and every dataset and pretrained model we use is publicly available.

Every registered gate, counted. Seventeen experiments carry a pre-registration written before the outcome was seen, with a hash of the analysis script recorded. Six met every gate they registered, both causal tests, both cascades, the Colored-MNIST replication and the CelebA operator arm. Eleven met some but not all. Each failure is reported where the result is used, and Table 3 lists them. The code release also holds registrations for further experiments this paper does not report. We rest no claim on a single gate. Counting every gate in its primary window, 137 of 179 pass, and with 179 tests a few would pass by chance. All generators, splits, protocols and hyperparameters are in Appendix B. Each registration is read-only and each outcome records the hash of the registration it tested. Appendix A states every claim against the settings it was tested in, including those where it was not tested and the one null replication.

REFERENCES

Alexander A. Alemi, Ian Fischer, Joshua V. Dillon, and Kevin Murphy. Deep variational information bottleneck. In *International Conference on Learning Representations*, 2017. URL `https://openreview.net/forum?id=HyxQzBceg`.

Jacob Andreas. Measuring compositionality in representation learning. In *International conference on learning representations*, volume 1, pp. 3, 2019.

Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization, 2020. URL `https://arxiv.org/abs/1907.02893`.

Chris Burgess and Hyunjik Kim. 3d shapes dataset. https://github.com/google-deepmind/3d-shapes/, 2018.

Giacomo Camposampiero, Pietro Barbiero, Michael Hersche, Roger Wattenhofer, and Abbas Rahimi. Scalable evaluation and neural models for compositional generalization. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL `https://openreview.net/forum?id=heQsyrMDzm`.

Rahma Chaabouni, Eugene Kharitonov, Diane Bouchacourt, Emmanuel Dupoux, and Marco Baroni. Compositionality and generalization in emergent languages. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 4427–4442, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.407. URL `https://aclanthology.org/2020.acl-main.407/`.

Ricky TQ Chen, Xuechen Li, Roger B Grosse, and David K Duvenaud. Isolating sources of disentanglement in variational autoencoders. *Advances in neural information processing systems*, 31, 2018.

Alexis Chevalier, Alexander Wettig, Anirudh Ajith, and Danqi Chen. Adapting language models to compress contexts. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 3829–3846, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.232. URL https://aclanthology.org/2023.emnlp-main.232/.

Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg, and Christopher Olah. Toy models of superposition. *Transformer Circuits Thread*, 2022. https://transformer-circuits.pub/2022/toy_model/index.html.

Tao Ge, Jing Hu, Lei Wang, Xun Wang, Si-Qing Chen, and Furu Wei. In-context autoencoder for context compression in a large language model. *arXiv preprint arXiv:2307.06945*, 2023.

Muhammad Waleed Gondal, Manuel Wüthrich, Djordje Miladinović, Francesco Locatello, Martin Breidt, Valentin Volchkov, Joel Akpo, Olivier Bachem, Bernhard Schölkopf, and Stefan Bauer. On the transfer of inductive bias from simulation to the real world: a new disentanglement dataset. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Julien Hendrickx, Alexander Olshevsky, and Venkatesh Saligrama. Graph resistance and learning from pairwise comparisons. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2702–2711. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/hendrickx19a.html.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. β-vae: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations (ICLR)*, 2017.

Phillip Isola, Joseph J. Lim, and Edward H. Adelson. Discovering states and transformations in image collections. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1383–1391, 2015. doi: 10.1109/CVPR.2015.7298744.

Huiqiang Jiang, Qianhui Wu, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. LLMLingua: Compressing prompts for accelerated inference of large language models. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 13358–13376, Singapore, December 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.825. URL https://aclanthology.org/2023.emnlp-main.825/.

Hyunjik Kim and Andriy Mnih. Disentangling by factorising. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2649–2658. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/kim18b.html.

Sébastien Lachapelle, Pau Rodriguez, Yash Sharma, Katie E Everett, Rémi Le Priol, Alexandre Lacoste, and Simon Lacoste-Julien. Disentanglement via mechanism sparsity regularization: A new principle for nonlinear ica. In *Conference on Causal Learning and Reasoning*, pp. 428–484. PMLR, 2022.

Qiyao Liang, Daoyuan Qian, Liu Ziyin, and Ila Fiete. Compositional generalization via forced rendering of disentangled latents. *arXiv preprint arXiv:2501.18797*, 2025.

Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Rätsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In *International Conference on Machine Learning (ICML)*, 2019.

Massimiliano Mancini, Muhammad Ferjad Naeem, Yongqin Xian, and Zeynep Akata. Open world compositional zero-shot learning. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5218–5226, 2021. doi: 10.1109/CVPR46437.2021.00518.

Loic Matthey, Irina Higgins, Demis Hassabis, and Alexander Lerchner. dSprites: Disentanglement testing sprites dataset. https://github.com/google-deepmind/dsprites-dataset, 2017.

Ishan Misra, Abhinav Gupta, and Martial Hebert. From red wine to red tomato: Composition with context. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1160–1169, 2017. doi: 10.1109/CVPR.2017.129.

Milton Montero, Jeffrey Bowers, Rui Ponte Costa, Casimir Ludwig, and Gaurav Malhotra. Lost in latent space: Examining failures of disentangled models at combinatorial generalisation. *Advances in Neural Information Processing Systems*, 35:10136–10149, 2022.

Milton Llera Montero, Casimir JH Ludwig, Rui Ponte Costa, Gaurav Malhotra, and Jeffrey Bowers. The role of disentanglement in generalisation. In *International Conference on Learning Representations*, 2021.

Jesse Mu, Xiang Li, and Noah Goodman. Learning to compress prompts with gist tokens. *Advances in Neural Information Processing Systems*, 36:19327–19352, 2023.

Muhammad Ferjad Naeem, Yongqin Xian, Federico Tombari, and Zeynep Akata. Learning graph embeddings for compositional zero-shot learning. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 953–962, 2021. doi: 10.1109/CVPR46437.2021.00101.

Nihal V Nayak, Peilin Yu, and Stephen H Bach. Learning to compose soft prompts for compositional zero-shot learning. *arXiv preprint arXiv:2204.03574*, 2022.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel HAZIZA, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=a68SUt6zFt. Featured Certification.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Florian Redhardt, Yassir Akram, and Simon Schug. Scaling can lead to compositional generalization. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=hZt0daVIZi.

Cinjon Resnick, Abhinav Gupta, Jakob Foerster, Andrew M. Dai, and Kyunghyun Cho. Capacity, bandwidth, and compositionality in emergent language learning. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '20, pp. 1125–1133, Richland, SC, 2020. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9781450375184.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10674–10685, 2022. doi: 10.1109/CVPR52688.2022.01042.

Lukas Schott, Julius von Kügelgen, Frederik Träuble, Peter Gehler, Chris Russell, Matthias Bethge, Bernhard Schölkopf, Francesco Locatello, and Wieland Brendel. Visual representation learning does not generalize strongly within the same domain. In *International Conference on Learning Representations (ICLR)*, 2022.

Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. Alfred: A benchmark for interpreting grounded instructions for everyday tasks. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10737–10746, 2020. doi: 10.1109/CVPR42600.2020.01075.

Naftali Tishby, Fernando C Pereira, and William Bialek. The information bottleneck method. *arXiv preprint physics/0004057*, 2000.

Arnas Uselis, Andrea Dittadi, and Seong Joon Oh. Does data scaling lead to visual compositional generalization? *arXiv preprint arXiv:2507.07102*, 2025.

Arnas Uselis, Andrea Dittadi, and Seong Joon Oh. Compositional generalization requires linear, orthogonal representations in vision embedding models. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=AQZZWVp6XA.

Thaddäus Wiedemer, Prasanna Mayilvahanan, Matthias Bethge, and Wieland Brendel. Compositional generalization from first principles. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Aron Yu and Kristen Grauman. Fine-grained visual comparisons with local learning. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 192–199, 2014. doi: 10.1109/CVPR.2014.32.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 11941–11952, 2023. doi: 10.1109/ICCV51070.2023.01100.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, et al. H2o: Heavy-hitter oracle for efficient generative inference of large language models. *Advances in neural information processing systems*, 36:34661–34710, 2023.

A CLAIM VERSUS EVIDENCE: WHAT WAS TESTED WHERE

This appendix states each claim of the paper against the settings where it was tested, including where it fails.

Table 2: **Where capture and the fix were tested.** *Left*, capture. The captured target named before training by the nearest seen combination, against chance, and the fraction of held-out cells whose prediction moves when that target is deleted and the model retrained, against re-seeding alone. *Right*, the fix. Sampled held-out composition at matched rate, plain → specialised read-out (Table 4).

capture	predicted / chance	deleted / re-seed
synthetic 6×6	0.654, code space[a]	1.00 / 0.55[b]
dSprites	0.761 / 0.071	1.00 / 0.300
Colored-MNIST	0.183 / 0.021	1.00 / 0.442
MPI3D-real (photographs)	0.750 / 0.036	1.00 / 0.075
DINOv2, SigLIP, CLIP	0.66–0.68 / <0.01[c]	1.00 / 0.11–0.33[d]
ALFRED goal text	0.602 / 0.018	0.911 / 0.327
Qwen2.5, frozen LM	0.214 / 0.028	0.958 / 0.875, **fails**
3DShapes, coarse hue	—	1.00 / 0.950[e]

fix	composition
synthetic 6×6	0.02 → 0.39
dSprites, location	0.14 → 0.65
dSprites, scale	0.00 → 0.35
Colored-MNIST	0.41 → 0.71
3DShapes	0.14 → 0.44
MPI3D (simulated)	0.28 → 0.35
ALFRED goal text	0.23 → 0.53

[a] pixel predictors are near chance. [b] vs. a matched non-target deletion. [c] CLIP and SigLIP, frozen encoders. [d] exploratory. [e] uninformative, re-seeding alone moves almost every cell.

Table 3 states the scope of every claim in one place, including the settings where a claim is *not* tested and the one setting where a replication is null. A reader should be able to see the boundary of the evidence without reconstructing it from the sections. One further registration for completeness: a generative comparison against β-VAE/FactorVAE/β-TCVAE returned uninterpretable (the baselines fail to reconstruct, 0.04–0.18); no claim rests on it. The registered fifth frozen-encoder cell (MPI3D, DINOv2) completed and is reported in Table 18 and Appendix H. The registered three-seed CelebA operator control has since landed *between* its two pre-declared outcomes and is reported in Appendix M.

Table 3: Every claim in the paper against the settings it was tested in. "Not tested" is not a synonym for "holds"; the rows marked synthetic-only are the paper's main coverage limitation and we say so in Section 7 as well.

claim	held in	not tested / did not hold
Capture: held-out predictions concentrate on one seen combination	dSprites, synthetic (n=118 cell×rate rows); concentration also on dSprites shape×position, Colored-MNIST (Arjovsky et al., 2020), MPI3D and coarse-hue 3DShapes (231 window rows); partly at LM scale (0.615)	**at LM scale an uncompressed control errs just as concentratedly**, so the concentration is not created by compression (Appendix I). The original 3DShapes hue grid has no capture window (the plain code composes until seen accuracy also collapses); a coarse-hue variant of the same dataset does
The target is predictable before training	dSprites (0.745 vs 0.071 chance), **MPI3D real camera images** (0.750 vs 0.036) and Colored-MNIST (0.183 vs 0.021), all from the nearest seen mean image, posterior-mean read-out	**which control fails is dataset-dependent**: on Colored-MNIST support degree ties the pixel predictor (0.218) because the pixel-nearest seen cell there always shares a factor value, and on MPI3D frozen ImageNet features predict, at 0.850 for their nearest-neighbour variant and 0.675 for the mean-image one in Table 9, as one expects of photographs, the dSprites finding that every control is at chance is specific to dSprites. On synthetic only the *code* predicts (0.654); pixel space is near chance, as that generator implies

claim	held in	not tested / did not hold
The relation is causal (delete-and-retrain, re-capture)	dSprites, 10 seeds, two rates, re-seed-controlled (β=0.01: 1.00 vs control 0.275 and a re-seed floor of 0.300, re-capture 0.750; β=0.1: 1.00 vs control 0.256 and a re-seed floor of 0.256, re-capture 0.564), synthetic ($40/40$ vs $22/40$), and **Colored-MNIST** (10 seeds, 1.00 $[1.00, 1.00]$ vs control 0.383 and a re-seed floor of 0.442; re-capture 0.117 against chance 0.021), and **MPI3D-real** (real photographs, 10 seeds, β=0.1: 1.00 $[1.00, 1.00]$ vs control 0.037 and a re-seed floor of 0.075 $[0.013, 0.138]$; re-capture 0.412 $[0.325, 0.512]$ against chance 0.037; paired 74:0; all four tests). Also on **frozen DINOv2, SigLIP and CLIP** embeddings of 3DShapes (relocation 1.00 every cell and seed against re-seed floors 0.11–0.33), though that run is *exploratory, unregistered and posterior-mean*, and it retrains the read-out over frozen features rather than from scratch (Appendix H). **Fails on coarse-hue 3DShapes**, where a re-seed alone relocates 0.950 of cells and the test is uninformative. **Cascade** to depth three on dSprites (5 seeds, 54 steps): relocation 1.00 at every depth, new target the nearest remaining seen cell in 0.704 $[0.59, 0.82]$ against 0.083 chance; random-k deletion 0.15–0.30, re-seed 0.35; and on **MPI3D-real** (5 seeds, 116 steps): relocation 1.00 at every depth, nearest remaining in 0.457 $[0.38, 0.57]$ against 0.038 chance, random-k 0.05–0.075, re-seed 0.075, 13 of 14 tests (pooled top-one bar of 0.5 missed)	**fails at LM scale**: re-drawing the seed without deleting anything already moves the prediction 0.875 against 0.958, so the target is not support-determined there
Composition degrades before in-support decodability	synthetic certified frontier, six frozen encoders, two LM families	—

claim	held in	not tested / did not hold
Not an information failure (InfoNCE certificate)	synthetic generator with added nuisance headroom	**partial on real images.** Fails on dSprites pixels, which lack labelled nuisance and so give the code no certifiable headroom. On **MPI3D real camera images** both the headroom gate and the *bits-are-there* gate pass: certified information permits joint accuracy 0.930, an upper bound on what the information permits that need not be attainable, where the code delivers 0.032. The certified sum still sits 0.126 nats below the Fano requirement, so one pre-registered criterion fails and we say *in part* (Appendix C)
Rate allocation is *not* the mechanism	synthetic 6×6 (1100 models) *and* DINOv2 / SigLIP / CLIP frozen embeddings of 3DShapes plus DINOv2 on MPI3D (4922 models, matched achieved rate)	one of four frozen cells is an exception (CLIP hue$\times$shape, $+0.011$ $[+0.004, +0.018]$). The from-scratch dSprites replication is underpowered (3 seeds) and null
Specialisation is graded, and can be learned	synthetic at three latent widths; frozen 3DShapes and UT-Zappos50K as a method vs AIN and β-VAE / FactorVAE / β-TCVAE (group sparsity recovers 92% of the routed gap on Zappos, ties it on 3DShapes, and beats all three rate-side baselines on both cells); factor-count misspecification: $F'{=}2$ peaks, $F'{=}1$ collapses	count is swept and an unknown F is never inferred; AIN, itself a specialising objective, ties group sparsity on Zappos and beats it on 3DShapes (0.354 vs 0.292)
The routed-vs-plain gap itself	positive mean gain on all seven benchmarks, 10-seed paired t intervals excluding zero on six, MPI3D positive in $7/10$ seeds but inconclusive (Table 4)	absolute numbers on real photographs remain low
Gist recombination in a frozen LM	Qwen2.5 at 1.5B and 0.5B, above zero in every seed	reported as an *existence* result only: the arms are not rate-matched (184.8 vs 136.1 nats) and the splice is ill-defined for a shared code
Support geometry predicts difficulty before training	UT-Zappos50K (pre-registered pass)	**met its bar on** 1 **of** 3: inverted on Fashion-Product-Images, right sign but under bar on MIT-States

B REPRODUCIBILITY DETAILS

Generators, splits, hyperparameters and protocols behind every experiment of Sections 3 to 6.

All experiments run on a single NVIDIA GB10 GPU with PyTorch 2.11. Seed counts are stated per result: the seven-benchmark battery (Table 4), the capture interventions, the mechanism factorial and the frozen-encoder sweeps all use 10 seeds, except the registered DINOv2–MPI3D cell, which uses 6. Any other exception is noted at the number itself. We never select hyperparameters on held-out combinations. Model selection uses a validation split of seen combinations only.

Which MPI3D split each result uses. Table 4 and the mechanism factorial use the *simulated* MPI3D split (the toy renders), which is the split the frozen-encoder sweeps were registered on. The capture concentration, the predictability test and the information certificate use **MPI3D-real**, the real

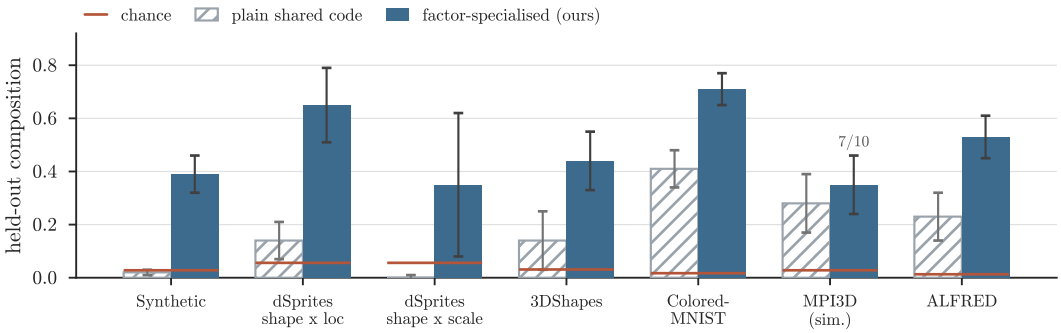

Figure 6: **Held-out composition for a plain shared code and a factor-specialised one on seven settings.** Plain (hatched) and specialised (solid) read-out at matched achieved rate, 10 seeds, sampled read-out, ±1 SD over seeds, chance ruled per group. Levels are plotted rather than the gap because the modest absolutes are part of the result.

Table 4: The battery of Figure 6 in full, including the posterior-mean difference (last column) that the figure omits. 10 seeds, sampled levels (Control 1). Bracketed intervals are 95% paired t intervals over the 10 seeds. They exclude zero on every benchmark except MPI3D, which is higher in 7 of 10 seeds and is positive but inconclusive. On frozen DINOv2 features of the same MPI3D images the registered factorial gives $+0.22$ $[+0.19, +0.25]$ over 6 seeds (Table 1). Shape × scale is the conservative dSprites pair.

			sampled read-out		paired, 10 seeds		
benchmark	domain	chance	plain VIB	routed	Δ sampled (95% t CI)	higher in	Δ mean
Synthetic (matched rate)	vectors	0.028	0.02 ± 0.01	$\mathbf{0.39 \pm 0.07}$	$+0.368$ $[0.319, 0.417]$	10/10	$+0.825$
dSprites, shape × location	image	0.056	0.14 ± 0.07	$\mathbf{0.65 \pm 0.14}$	$+0.514$ $[0.435, 0.594]$	10/10	$+0.536$
shape × scale	image	0.056	0.00 ± 0.01	$\mathbf{0.35 \pm 0.27}$	$+0.347$ $[0.159, 0.535]$	10/10	$+0.360$
3DShapes	image	0.031	0.14 ± 0.11	$\mathbf{0.44 \pm 0.11}$	$+0.300$ $[0.257, 0.342]$	10/10	$+0.336$
Colored-MNIST	image	0.017	0.41 ± 0.07	$\mathbf{0.71 \pm 0.06}$	$+0.292$ $[0.247, 0.337]$	10/10	$+0.200$
MPI3D (sim. split)	robot image	0.028	0.28 ± 0.11	$\mathbf{0.35 \pm 0.11}$	$+0.069$ $[-0.007, 0.145]$	7/10	$+0.076$
ALFRED	goal text	0.013	0.23 ± 0.09	$\mathbf{0.53 \pm 0.08}$	$+0.303$ $[0.237, 0.370]$	10/10	$+0.302$

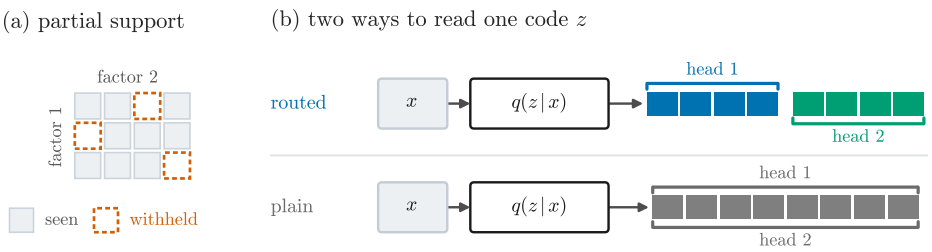

Figure 7: **The setup.** Training uses a partial support and composition is accuracy on the withheld cells **(a)**. One encoder and one rate budget are read two ways **(b)**: a *routed* arm wires each head to a random disjoint half of the latent dimensions; a *plain* arm lets both heads read all of z; everything else is identical.

camera images, and every result quoted as "real camera images" in the main text is from that split. The two are never pooled.

The gap metric. With $C = \prod_f V_f$ candidate combinations, raw held-out accuracy on some settings sits near the chance floor $1/C$, which hides the effect. For those settings we report a *gap*: $\text{gap} = s(z^\star) - s(z_{\text{rand}})$, where $s(\cdot)$ is held-out accuracy, $z^\star$ is the correctly composed token, and z_{rand} is a structureless control drawn from the prior at the same scale. The control is fixed in advance so it cannot be retuned per benchmark, and we use the gap only where raw accuracy is at the floor; every other number in the paper is raw joint held-out accuracy.

The nuisance-headroom task. The flagship dissociation cannot be run on the base grid, because there $\sum_f H(y_f) = \ln 36$ coincides with the largest factor information any code can carry, so no operating point can be certified above the threshold. We therefore add two independent nuisance

factors of six values each, resampled independently of the two label factors, which raises the achievable $I(Z;X)$ without changing $\sum_f H(y_f) = 2\ln 6 = 3.584$ nats. Each sample concatenates one-hot vectors for the two label and two nuisance factors and applies the fixed random generator $x = \tanh(oW_1)W_2 + 0.3\eta$, where $o \in \{0,1\}^{24}$, $W_1 \in \mathbb{R}^{24\times128}$, $W_2 \in \mathbb{R}^{128\times64}$, and $\eta \sim \mathcal{N}(0, I)$. The encoder has a $64 \to 128 \to 16$ Gaussian bottleneck, two label heads, and two nuisance heads; nuisance cross-entropy has weight 0.3. Held-out cells are drawn as in the base protocol, $\max(F, \lceil V^F/V \rceil) = 6$ of the 36 label combinations, leaving 30 seen, subject to every factor value appearing in the seen support; the conjunction probe is therefore a 36-way probe trained on the 30 seen classes. Training runs 2500 steps, the probe 1200, the InfoNCE critic 1000 and the conjunction probe 800, over 5 seeds and $\beta \in \{0.09, 0.12, 0.16, 0.20, 0.25, 0.30\}$, with the operating point fixed at β=0.12 from a pilot before the confirmatory run. Mutual information is estimated on fresh samples from the theorem's own test law, the uniform full product. Note that the split differs from the base task, which holds out 8 of 36: numbers from the two are not interchangeable.

Synthetic compositional task. Inputs are dimension 64, generated as $x = \tanh(\text{onehot}(y)\, W_1)\, W_2 + 0.3\,\eta$ with fixed random W_1, W_2 and $\eta \sim \mathcal{N}(0, I)$. Two factors, shape and colour, each with 6 values, and 8 of the 36 combinations are held out. The encoder is a single hidden layer of width 128 with ReLU, followed by linear maps to a 16 dimensional mean and log-variance. For the routed arm the token is split into two blocks of width 8, each with its own linear head. The decorrelation weight is $\lambda = 1.0$. Training uses Adam at learning rate 2×10^{-3}, batch size 256, for 3000 steps, with β swept for the rate axis.

Image benchmarks. dSprites (737k images of 64×64) uses shape (3 values) and scale (6 values). 3DShapes (480k images of $64 \times 64 \times 3$) uses shape (4 values) and scale (8 values). MPI3D (1.04M images of $64 \times 64 \times 3$) uses object colour (6 values) and object shape (6 values). Colored-MNIST (60k MNIST digits colourised to $28 \times 28 \times 3$ with the six RGB/CMY corners) uses digit (10 values) and colour (6 values). In every case the remaining generative factors vary freely as nuisance variation. The encoder is four convolutional layers (32, 32, 64, 64 channels, kernel 4, stride 2, padding 1) with ReLU (three kernel-3 layers for the 28×28 Colored-MNIST), flattened to a linear layer of width 128, then linear maps to a 16 dimensional mean and log-variance. The routed arm splits this into two blocks of width 8 with per-factor heads. Training uses Adam at learning rate 10^{-3}, batch size 256, decorrelation weight λ up to 10, for 4000 steps (8000 for the tuned dSprites run, 12000 for MPI3D), sweeping $\beta \in \{0, 10^{-3}, 10^{-2}, 10^{-1}\}$ and selecting on a seen-only validation split (never on held-out combinations). Held-out fractions are 4 combinations (dSprites), 8 (3DShapes), 8 (MPI3D), and 12 (Colored-MNIST), drawn at random per seed so each value still appears in other combinations. This applies to the MPI3D row of Table 4 as well: both its accuracy column and its paired interval come from the ten-seed protocol in which each seed draws its own held-out split and shares it across arms and read-outs.

Task language (ALFRED). We pair ALFRED (Shridhar et al., 2020) goal descriptions with their sub-goal (object, receptacle) factors (10 objects $\times$ 8 receptacles, n=2347 episodes, chance 0.0125). The goal text is tokenised and encoded by a word embedding (dimension 64) followed by a GRU (hidden 128). Its final state maps to a 16 dimensional VIB token (mean and log-variance) split into object and receptacle blocks of width 8 with per-factor heads and the decorrelation penalty (λ=1). Training uses Adam at learning rate 2×10^{-3}, batch size 128, 3000 steps, sweeping $\beta \in \{0, 10^{-3}, 10^{-2}, 10^{-1}\}$. We hold out (object, receptacle) pairs whose object and receptacle each appear in other pairs, and report held-out joint accuracy (mean over 10 seeds), giving the plain and routed entries of Table 4 (paired $\text{routed} - \text{plain} = +0.303$, 95% CI $[0.246, 0.361]$, higher in all 10 seeds; the posterior-mean read-out of the same runs gives $+0.30$ $[0.23, 0.38]$).

Capture on ALFRED goal text. The frozen-decoder null of Section 6 leaves a question its own wording raises: is the boundary *language*, or is it the frozen decoder? ALFRED separates them. It is language, but the shared read-out sits on a *trained* encoder, which is the setting the mechanism is stated for. We ran the three findings there, each pre-registered, and each with an explicit stop condition.

The floor first. The causal test is unreadable wherever retraining alone moves the prediction about as often as deleting the target does, which is what happens at language-model scale (0.875 against 0.958). Holding the split and support fixed and redrawing only the training seed, the ALFRED floor

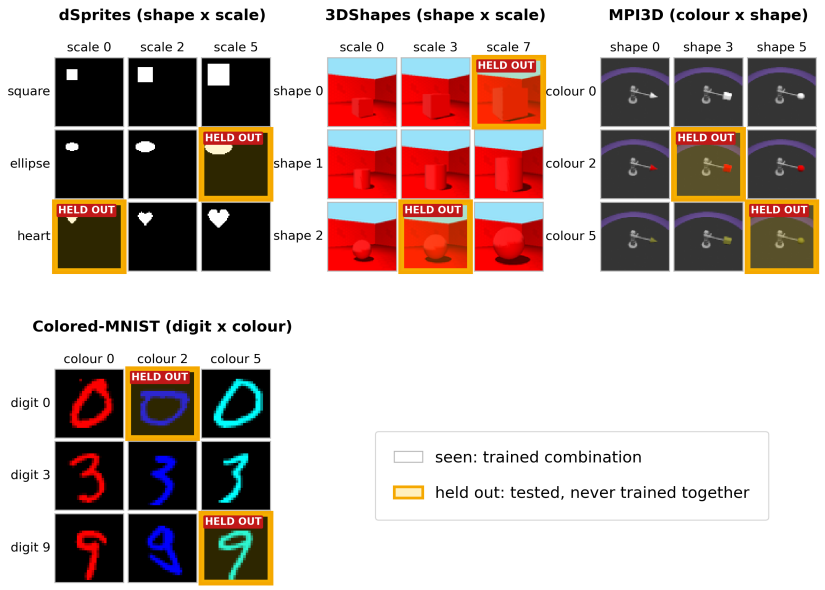

Figure 8: **Held-out-combination protocol.** Rows and columns are the two generative factors of each benchmark. Faded cells are *seen* combinations used to train the representation. The bright, gold-framed cells tagged HELD OUT are combinations never seen together in training, on which we test compositional generalisation (legend, bottom). Four image benchmarks (dSprites, 3DShapes, MPI3D, Colored-MNIST) appear in Table 4.

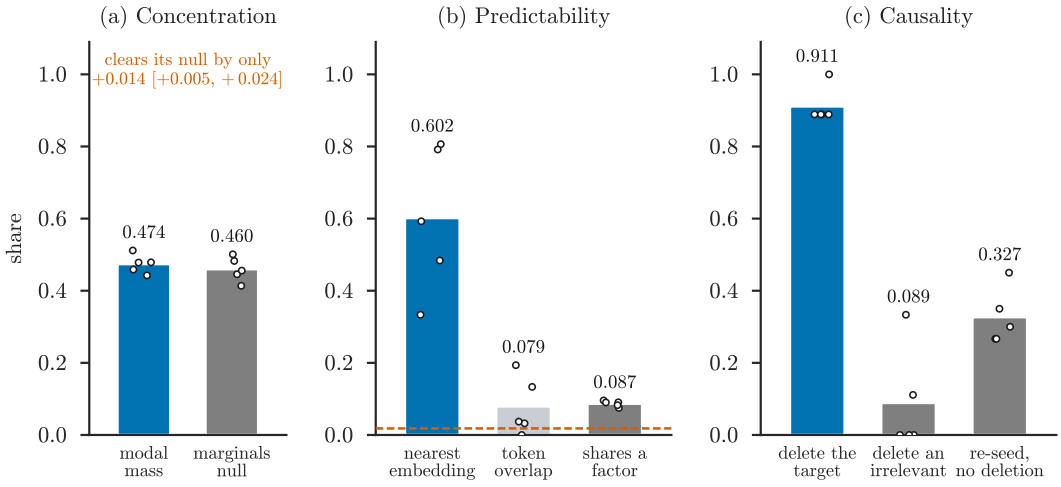

Figure 9: **The three capture findings on ALFRED goal text**, each beside its own control. Bars are means over 5 splits, dots are the individual splits, read-out sampled throughout. **(a)** Concentration clears its product-of-marginals null by only $+0.014$ $[+0.005, +0.024]$, and one split falls below it, so the *amount* is not claimed here. **(b)** The seen pair whose mean frozen sentence embedding is nearest names the captured target at 0.602, against 0.087 for picking uniformly among seen pairs sharing an object or a receptacle (dashed rule: the 0.018 uniform chance rate). Token overlap sits *below* the shared-value null, so surface form carries nothing. **(c)** Deleting the captured pair relocates the prediction 0.911 of the time, against 0.089 for deleting an equally sized irrelevant seen pair and 0.327 for redrawing the training seed with no deletion. The floor is what makes this panel readable; at language-model scale it is 0.875 and the same test says nothing.

is 0.3267 $[0.2733, 0.3933]$, comparable to dSprites' 0.300. The deletion arms were registered only after this was measured (Figure 9).

Predictability. Among captured pairs, the seen pair whose mean frozen sentence embedding is nearest names the target at 0.602, against 0.087 for picking uniformly among seen pairs that share

an object or a receptacle, and 0.018 chance (paired difference over the shared-value null $+0.515$ $[+0.353, +0.677]$; 5 splits $\times$ 4 training seeds). This is the gate the language-model arm failed, where embedding-nearest reached 0.214 against a 0.165 shared-value null and factor sharing explained nearly all of it. Token overlap scores 0.079, *below* the null, so surface form carries nothing: the address is carried by semantic similarity, as pixel distance carries it on images.

Causality. Deleting the captured pair from the training support and retraining relocates the prediction 0.911 of the time, against 0.089 for deleting an equally sized seen pair that is nobody's target, and the 0.3267 floor above ($+0.822$ $[+0.733, +0.889]$ over the control, $+0.584$ $[+0.562, +0.629]$ over the floor; 99 models, 5 splits $\times$ 3 targets $\times$ 3 training seeds). Deletion does not force the move: removing the pair (o, r) leaves both o and r in other pairs, so the two heads can still emit that combination. They stop doing so.

What does not hold, and what we did not test. Concentration only *marginally* clears its product-of-marginals null, $+0.0142$ $[+0.0046, +0.0240]$, with one split of five negative, so we do not claim the amount here. Re-capture onto the next-nearest remaining pair is *untested*: text has no established distance ordering to walk, and inventing one after seeing the outcome is exactly what the registrations exist to prevent. Read-out is sampled throughout.

Disentanglement baselines. β-VAE, FactorVAE, and β-TCVAE share the synthetic encoder above plus a symmetric decoder, trained unsupervised on seen combinations. We then freeze the mean code and fit linear per-factor probes for 1500 Adam steps before measuring held-out composition. FactorVAE uses a density-ratio discriminator for the total-correlation term, and β-TCVAE uses the minibatch-weighted sampling estimator. These are faithful reimplementations of the published objectives. The attribute-invariant-network baseline follows the split-path / per-attribute-head design of Camposampiero et al. (2025).

VLM-supplied factor labels without human-provided per-image labels. The per-factor labels the routed arm needs can also come from a vision-language model rather than a human. On Colored-MNIST we label each training image with Qwen2.5-VL-7B (prompted for the digit and the colour, with a one-line colour description, and parsed from its reply), use no human-provided per-image labels, and train the routed arm on the VLM labels alone. The labels reach 0.93 joint accuracy (digit 0.93, colour 0.99, read successfully on all images), and a routed code trained on them reaches 0.69 ± 0.06 held-out composition, rising to 0.76 ± 0.04 with a self-cleaning pass that drops the highest-loss 15% of labels and retrains. This is comparable to, and after self-cleaning slightly exceeds, the 0.71 ± 0.02 true-label baseline under the same 1440-image protocol (3 seeds), so factor labels can be sourced automatically on readable images with essentially no penalty. This supplies the supervision. The partition is still specified.

C THE INFORMATION CERTIFICATE

This appendix supports Section 4.4, that capture is not an information failure.

The factors are readable through the model's own heads. Read through the *same* trained heads of the *same* plain dSprites code, a composition operator built from per-factor prototypes recovers held-out combinations at 0.858 $[0.755, 0.961]$ over 30 seeds, while a real held-out image pushed through that encoder scores 0.020 $[0.000, 0.044]$. The operator agrees to three decimals with a closed-form generalised-least-squares read-out (Theorem 1) and beats the naive additive rule by $+0.117$ $[+0.023, +0.211]$. On MPI3D-real the held-out-conditioned code retains 2.06 $[1.79, 2.33]$ of 2.94 nats, and Fano then permits joint held-out accuracy of 0.80 $[0.78, 0.83]$.

The certificate on real camera images, and what fails there. The dSprites failure above is a protocol gap: that grid carries no labelled nuisance factor, so $\sum_f H(y_f)$ coincides with the most factor information any code can carry and no operating point can be certified above threshold. We therefore repeated the *synthetic* recipe on MPI3D **real camera images**, taking object colour $\times$ object shape as the label pair, so that $\sum_f H(y_f) = 2 \ln 6 = 3.584$ nats exactly as on the synthetic task, and attaching nuisance heads at the synthetic weight (0.3) to the five free labelled factors. At a pre-registered β=0.03, chosen by a declared rule from a pilot, over 5 seeds: the InfoNCE lower

bound on $I(Z;X)$ is 4.29 $[4.16, 4.42]$ nats against the 3.584-nat scale, and it is not estimator-limited, sitting 1.95 nats below the $\ln(\text{batch}) = 6.24$ ceiling with a batch-2048 repeat agreeing to within 0.27 nats. Seen-conjunction accuracy is 0.960 while held-out composition is 0.027, at chance $(1/36)$. The information-*headroom* condition therefore transfers to real photographs, and this is the gate that dSprites failed.

The label-side test first appeared to fail, and the cause was an instrument artefact. The bits were there. With the probe fitted on the *seen* support, the variational label bound goes vacuous at a capture operating point: colour contributes a cross-entropy of 0.049 nats while shape contributes 2.362, above $\ln 6$ and hence nothing. Shape is large only because on held-out cells that probe is *confidently wrong*, at 9–14 nats, and cross-entropy is unbounded above. That failure mode is capture itself, a property of the probe that the code does not share.

The variational bound $I(Z;y_f) \geq H(y_f) - \mathbb{E}_p[-\log q]$ holds for *any* q: only the expectation must be taken under the certified law. Fitting the probe on the same uniform full-product law the bound is evaluated under, which is the synthetic certificate's own protocol (Appendix C), removes the vacuity on real camera images. Shape cross-entropy falls to 0.358 $[0.336, 0.380]$ nats, five times below $\ln 6$, in five seeds of five, and $\sum_f I(Z;y_f) = 3.206$ $[3.179, 3.233]$ nats, 89.5% of the 3.584-nat ceiling. That certified information permits a joint accuracy of 0.930 $[0.924, 0.936]$ (Fano bounds what an information shortfall can explain and has no achievability half), where the code delivers 0.032, so the pre-registered *the bits are there* gate passes on real photographs. This fit uses held-out cells and therefore measures what the code carries, a separate question from composition: every composition number in this paper comes from probes fitted without held-out labels.

One criterion still fails, so we continue to say *in part*. The certified sum sits 0.126 $[0.095, 0.158]$ nats below the Fano requirement at the model's own seen-support error, and the pre-registered threshold asks the bound to come within 0.051 nats of the ceiling. We note without relying on it that the synthetic flagship does not clear that threshold either.

Conditioning on the held-out cells. A fair objection to any full-product-law accounting is that a code which identifies every seen cell perfectly and maps each held-out cell onto a seen one is itself nearly information-complete under that law, since 30 of 36 cells are seen. The accounting that answers it conditions on the held-out cells alone. Using the label probe of the preceding paragraph (fitted on the full-product law, held-out labels included, so this measures what the code *carries* and is not a composition test), we evaluate its cross-entropy on the 8 held-out cells only, at the certificate's registered operating point $\beta{=}0.03$, five seeds. Per factor, $I(Z;y_f \mid \text{held}) \geq H(y_f \mid \text{held}) - \text{CE}_f$ gives 1.50 of 1.53 nats for colour and 0.56 of 1.40 for shape, 2.06 $[1.79, 2.33]$ of 2.94 in total. Fano per factor on the held law permits colour accuracy 0.996 and shape accuracy 0.79 $[0.77, 0.82]$. For the joint we take the conservative route, $I(Z;Y \mid \text{held}) \geq \sum_f I(Z;y_f \mid \text{held}) - I(y_c;y_s \mid \text{held})$ (the two factors are dependent on an 8-cell law), which still permits joint held-out accuracy 0.80 $[0.78, 0.83]$ against 0.03 $[0.00, 0.06]$ delivered by the model's own heads: permitted minus delivered $+0.77$ $[+0.75, +0.80]$. These are cross-entropy upper bounds on conditional entropy, valid for any probe, so the plug-in estimator bias discussed above does not enter. What it does not show is that a read-out trained on the seen support could use those bits; that is the placement question the rest of the paper is about.

The certificate has two parts. A converse gives how much information reliable recovery requires, and a measurement gives how much information the code carries. The dissociation needs both, and together they show that lost information does not explain the failure.

C.1 The converse: how much rate reliable recovery needs

Proposition 2 (Information necessary for reliable factor recovery)**.** *Let y be uniform on the product $\prod_f [V_f]$ with every $V_f \geq 2$ (the composition-test distribution) and let $Z{=}(Z_1, \ldots, Z_F)$ be a possibly stochastic code whose per-factor decoders satisfy $\Pr[\hat{y}_f{\neq}y_f \mid y{=}c] \leq \delta \leq \frac{1}{2}$ for every combination c. With routed decoders $\hat{y}_f{=}g_f(Z_f)$,*

$$\textstyle\sum_f I(Z_f;X) \;\geq\; \sum_f I(Z_f;y_f) \;\geq\; \sum_f H(y_f) - \sum_f \big(h_{\text{b}}(\delta) + \delta \log V_f\big),$$

Table 5: **Flagship nuisance-headroom test** (pre-specified $\beta = 0.12$, five paired seeds; mean $\pm 95\%$ t-interval). Independent nuisance factors let $I(Z;X)$ exceed the zero-error factor scale $2\ln 6 = 3.584$ nats. The InfoNCE lower bounds remain above that scale in both arms. The same factor heads give nearly equal sampled seen accuracy, yet plain held-out composition is 0.604 while routing reaches 0.843; the paired gain is $+0.239 \pm 0.104$ and positive in every seed. The routed arm's InfoNCE bound is 0.361 ± 0.153 nats lower, which we report as a fact about the two bounds and not as an ordering of the underlying quantities, since one lower bound falling below another orders nothing. As a separate check, a 36-output probe trained and evaluated over the 30 seen conjunction classes scores 0.950 ± 0.010 and 0.952 ± 0.003. The complete pre-specified rate frontier is reported below. A companion run over the same models reports 3.40 nats where this table reports 3.35: a systematic estimator difference. This table stores a variational probe *lower bound*, the companion an MC *plug-in*, and the bound is lower in 45 of 60 matched cells. We quote the bound as the conservative one; KL and InfoNCE agree within MC noise.

arm	achieved KL	InfoNCE $I(Z;X)$ LB	seen same-head	held-out same-head
plain shared	8.04 ± 0.33	4.99 ± 0.12	0.943 ± 0.012	0.604 ± 0.078
factor-routed	7.80 ± 0.15	4.63 ± 0.08	0.937 ± 0.003	$\mathbf{0.843 \pm 0.055}$

where h_b is the binary entropy, and the same lower bound holds for the joint rate $I(Z;X) \geq I(Z;y)$ under unrouted decoders. As $\delta \to 0$ it tends to $\sum_f H(y_f)$; for finite δ it gives a graded *converse, consistent with the observed ramp.*

Proof. All quantities are under the composition-test law: $y \sim \mathrm{Unif}(\prod_f [V_f])$ (so the y_f are independent and uniform, $H(y_f) = \log V_f$), $X \sim p(x \mid y)$, and $Z \sim q(z \mid x)$ read from X alone. *(i, routed).* By Fano, for each f, $H(y_f \mid Z_f) \leq h_b(P_e^f) + P_e^f \log(V_f - 1)$ with $P_e^f = \Pr[\hat{y}_f \neq y_f]$; since $P_e^f \leq \delta \leq \frac{1}{2}$, $p \mapsto h_b(p) + p\log(V_f - 1)$ is increasing on $[0, \frac{1}{2}]$, and $\log(V_f - 1) \leq \log V_f$, we get $H(y_f \mid Z_f) \leq h_b(\delta) + \delta \log V_f$, hence $I(Z_f; y_f) \geq H(y_f) - h_b(\delta) - \delta \log V_f$. As $y_f - X - Z_f$ is Markov (given X the encoder block is independent of the label), data processing gives $I(Z_f;X) \geq I(Z_f;y_f)$; sum over f. *(ii, joint).* For an unrouted decoder $\hat{y}_f = g_f(Z)$, the same Fano step gives $H(y_f \mid Z) \leq h_b(\delta) + \delta \log V_f$. Independence of the y_f under the test law and subadditivity of conditional entropy then give $I(Z;y) = H(y) - H(y \mid Z) \geq \sum_f [H(y_f) - h_b(\delta) - \delta \log V_f]$; and $I(Z;X) \geq I(Z;y)$ by data processing on $y - X - Z$. When blocks are conditionally independent given X, $\sum_f I(Z_f;X) - I(Z;X) = \mathrm{TC}(Z_1, \ldots, Z_F)$, so the routed and joint rates differ by the blocks' total correlation, which a decorrelation penalty suppresses; for deterministic discrete codes $I(Z_f;X) = H(Z_f)$, connecting the routed and joint forms of the same converse. □

□

C.2 Measuring the information the code carries

The variational rate β controls the KL term, an upper bound on $I(Z;X)$, and models are read out at the posterior mean, so the achieved KL overstates the information the code actually uses. We lower-bound $I(Z;X)$ with InfoNCE (separable critic, batch 512, ceiling $\ln 512 = 6.24$ nats, stochastic Z). The estimate is well behaved: $I(Z;X) \leq$ KL everywhere (data processing), monotone in β, and below the ceiling. Table 6 shows the achieved KL runs 8–13 nats where the InfoNCE bound is only ≈ 3.5; sampled composition co-varies with this lower bound, while the mean read-out masks the degradation. Because all displayed InfoNCE estimates are at or below 3.584, this base-grid table does not establish a crossing of the zero-error scale. A separate sweep does establish the *law* across grids. Varying the factor count $F \in \{2, 3, 4\}$ and the per-factor alphabet from 3 to 12 (414 trained models over 23 configurations) and locating each configuration's collapse point, the collapse point tracks $\sum_f H(y_f)$ at Pearson $r = 0.91$ (strong-probe estimate) and $r = 0.95$ (Monte-Carlo plug-in) over the 14 configurations that collapse inside the swept range, while the *achieved-KL* proxy, the quantity a practitioner would reach for, tracks it at $r = 0.21$. The threshold is therefore set by the factor information a code must carry, which the budget it is charged barely tracks, and it holds as the number of factors changes, beyond the 6×6 grid of Table 6. The slope is 0.56–0.61, below 1, so we read it as a graded threshold that predicts where collapse happens, with a loose fit. Of the main text, the seven-benchmark battery (Table 4) reports *both* currencies on the same trained models: its plain and routed columns and the paired interval are sampled, with the posterior-mean difference

given alongside for comparison. The capture and factorial sections likewise read sampled codes, and the sampled read-out is the stricter measure used for this information calibration. The collapse conclusion is robust to this choice: at the scarce-rate end plain composition reaches 0.00 under both read-outs (Table 6), which diverge only at intermediate rates.

Table 6: An InfoNCE lower bound on mutual information vs. achieved KL (plain token, synthetic 6×6, 3 seeds). Achieved KL overstates the code's information by 1.1–3.6×, most at loose rates. Sampled composition co-varies with the information lower bound rather than KL; this is a calibration and leaves sufficiency untested.

β	0.003	0.03	0.1	0.3	1.0
InfoNCE $I(Z;X)$ (nats)	3.54	3.49	3.36	2.92	0.85
achieved KL (nats)	12.9	8.4	6.2	4.0	0.9
held-out comp. (sampled)	0.84	0.73	0.50	0.18	0.00
held-out comp. (mean)	0.99	0.96	0.92	0.64	0.00

The base grid alone cannot establish information headroom because its maximum factor information equals the zero-error scale. We therefore use the dedicated nuisance-headroom experiment below as the scope test; unlike earlier seen-support diagnostics, it evaluates $I(Z;X)$ on fresh samples from the uniform full-product law used by Proposition 2.

What the headroom run does and does not establish. The full six-rate frontier was fixed before the five-seed run; $\beta = 0.12$ was pre-specified from a shorter pilot rather than selected by held-out accuracy. At that point the plain code has an InfoNCE lower bound $I(Z;X) = 4.99 \pm 0.12$ nats, whose entire seed-level 95% interval is above $2 \ln 6 = 3.584$. The model's own factor heads score 0.943 ± 0.012 jointly on seen inputs, while the same heads compose held-out inputs at 0.604 ± 0.078. The routed arm has similar achieved KL (7.80 ± 0.15 vs. 8.04 ± 0.33) and seen accuracy (0.937 ± 0.003), yet composes at 0.843 ± 0.055. The paired gain is $+0.239 \pm 0.104$ and positive in all five seeds (Table 5). This is direct evidence of *differential degradation* above the necessary joint-information condition. It is not the stronger $0.86/0.18$ base-grid probe separation: that point is on the base grid and its measured InfoNCE bound is approximately 2.9 nats. We therefore use the headroom run for the strong claim and the base sweep only to show the limiting regime.

InfoNCE protocol. A fixed critic has separate two-hidden-layer MLPs ($D_X \to 256 \to 256 \to 64$ and $D_Z \to 256 \to 256 \to 64$, ReLU) and scores pairs by dot product. It is trained for 1000 Adam steps at learning rate 5×10^{-4} with symmetric $X \leftrightarrow Z$ cross-entropy, batch size 512, and gradient norm clipped at 5. The reported bound is then evaluated without fitting on 20 fresh batches (10,240 pairs) drawn uniformly from all 36 factor combinations with nuisance values independently resampled. The 36-output conjunction probe is supplementary: it is trained and evaluated only over the 30 seen classes.

Label-information lower bound. Independently of InfoNCE, a probe $Z \to 256 \to 256$ with one six-way head per label factor is trained for 1200 Adam steps at learning rate 10^{-3} on posterior samples from the same uniform full-product law (batch 512). Cross-entropy is evaluated on 20 fresh batches and converted factorwise using the variational bound $I(Z;y_f) \geq H(y_f) - \mathrm{CE}_f = \ln 6 - \mathrm{CE}_f$; we report $\sum_f \max\{0, \ln 6 - \mathrm{CE}_f\}$. At the pre-specified $\beta = 0.12$ point this bounded probe gives 3.349 ± 0.044 nats for the plain arm and 3.379 ± 0.029 for the routed arm (mean $\pm 95\%$ CI, five seeds). This is a lower bound on label-side information. It is neither an equality nor a certificate that either code exceeds the 3.584-nat zero-error scale.

The gap is not residual information loss. A reader can still ask whether the plain arm's held-out failure reflects the small shortfall between its factor information and the ceiling. It does not, and the margin is not close. Extending the frontier to $\beta = 0.03$ certifies the point more strongly: the InfoNCE lower bound reaches 5.826 nats (interval lower limit 5.797), so $I(Z;X)$ is certified 2.21 nats *above* $\sum_f H(y_f)$, while $\sum_f I(Z;y_f) = 3.529$ sits at 98.5% of its ceiling, a deficit of 0.055 nats. Sampled seen-conjunction decodability is 0.993 and sampled held-out composition is 0.614 ± 0.114. We emphasise that the pre-specified point on this frontier is β=0.12, where the same pattern holds a little less sharply (certified 1.29 nats above the scale under the same interval-lower-limit convention, 1.41 at the mean; seen conjunction 0.950, held-out 0.604, both from this table's canonical run); β=0.03

was added afterwards because it certifies more strongly, and we label it exploratory. It should be read with its estimator in view: the factor-information estimate at this point is an MC mixture-of-Gaussians plug-in whose audited residual bias is $+0.05$ to $+0.13$ nats, larger than the 0.055-nat deficit it would certify, so this point corroborates the pre-specified one and cannot stand alone. Now bound how much of the gap a residual information shortfall could explain. The encoder's measured conjunction entropy *on the held-out cells* (the conditional entropy restricted to the held-out conjunction; the margin below is $6\times$ under either reading) is within 0.055 nats of complete, and Fano's converse turns that into an error floor near zero: the information the code carries does not *force* an error anywhere near the observed 0.386. We are careful about the direction. Fano bounds error from below and has no achievability half, so this does not certify that 0.99 is attainable by any decoder; converting a small conditional entropy into a small *achievable* error needs a different tool, and we do not claim it. What the converse does establish is the negative that matters here: a 0.055-nat deficit cannot be the cause of a 0.386 gap, so what is missing is not information. At this same operating point the two arms carry indistinguishable certified information (5.83 against 5.84) and indistinguishable decodability (0.993 against 0.995), and compose at 0.614 against 0.941. Below the crossing the picture inverts as the necessary condition requires: once $I(Z;X)$ falls under 3.584, sampled seen decodability itself drops below 0.90 and keeps falling to 0.15, and in that regime the information itself is lost.

Operator or representation? Protocol for the main-text comparison. Every operator is read through the *same* frozen heads of the *same* trained plain code: dSprites shape $\times$ scale, $Z{=}16$, $\beta{=}0.01$, 4000 steps, 30 seeds $\times$ 4 held-out cells (117 identifiable). Prototypes are per-factor means over the seen support. Decoding *actual* held-out images with no operator gives 0.00 for the plain token against a routed arm that is higher in all 10 seeds on the conservative pairing (paired $+0.36$, 95% CI $[0.19, 0.53]$). A widened $n_{\text{held}}{=}7$ replication over 20 seeds reproduces the ordering. This is also why we report generative composition under arm-appropriate operators (Appendix M) rather than under block-swap alone.

D SPLIT-GEOMETRY SPREAD

This appendix supports the point of Section 3 that absolute composition depends on which combinations a seed withholds. It collects every dSprites operating point and the per-seed spread.

All dSprites operating points. Several dSprites held-out numbers appear across the paper. Table 7 collects them with their protocol. They are the same routed method under different rates and supervision, and all are reported. The primary dSprites result is the separable shape $\times$ location pair: 0.65 ± 0.14 sampled over 10 seeds (Table 4), 0.69 ± 0.16 at the posterior mean on the same models, against 0.64 ± 0.17 in the earlier five-seed reference run. The entangled shape $\times$ scale pair is the harder, conservative case, and we report it by its paired test: $+0.36$ $[0.19, 0.53]$ over 10 seeds, higher in every one.

Table 7: All dSprites held-out composition values in one place. Same method throughout, showing the conservative shape $\times$ scale case by its paired test (the primary dSprites result is shape $\times$ location 0.65). Other rows vary the rate or the supervision, or report the paired test.

setting	held-out
Plain VIB (fixed protocol)	0.00
routed over plain, paired, 10 seeds (**conservative**)	$+\mathbf{0.36}$ (all 10)
routed at a looser rate ($R \approx 25$)	0.50
routed, matched-achieved-rate frontier peak ($R \approx 22$)	0.80
routed, combination-only labels	0.60
AIN reimplementation	0.24

dSprites seed variance (shape $\times$ scale). On the conservative shape $\times$ scale pairing the dSprites routed arm has high seed variance, with per-seed held-out accuracies of $0.93, 0.49, 0.26, 0.12, 0.01$ (seeds 0 to 4) against a plain token that is exactly 0.00 on held-out combinations at every rate and seed. We therefore report this cell by its paired test rather than by a mean: over 10 seeds the paired difference $\text{routed} - \text{plain}$ is $+0.36$ with a 95% CI of $[0.19, 0.53]$ and the routed arm higher in all 10

seeds. On MPI3D over 10 seeds the paired difference is $+0.08$ (95% CI $[0.01, 0.15]$, routed higher in 8), so the advantage is directional on both high-variance cases despite the marginal spread. The same 10-seed paired test on the remaining two image benchmarks gives $+0.34$ (95% CI $[0.25, 0.42]$) on 3DShapes and $+0.20$ (95% CI $[0.15, 0.25]$) on Colored-MNIST, with the routed arm higher in all 10 seeds on each, so all four image benchmarks show a paired advantage with a 95% CI above zero. The routed arm exceeds the plain token in every seed. The variance reflects that some seeds reach near-perfect held-out composition and others only partial compositional recovery. The spread is driven by *which* combination is held out: within a fixed held-out split independent inits cluster, while the split moves the mean from $\sim$0.5 to $\sim$0.1 (held-out combinations are drawn per seed, so seed variance conflates split difficulty with init). Averaging inits cannot remove this. An ensemble of five routed models gives 0.32 ± 0.18 across splits, no better than the single-model mean, because the variance comes from the factor geometry of which combinations are composable.

E CAPTURE: FULL TABLES

The full capture results behind Section 4, the concentration, the before-training predictor, the causal test with its re-seed floor, and the cascade.

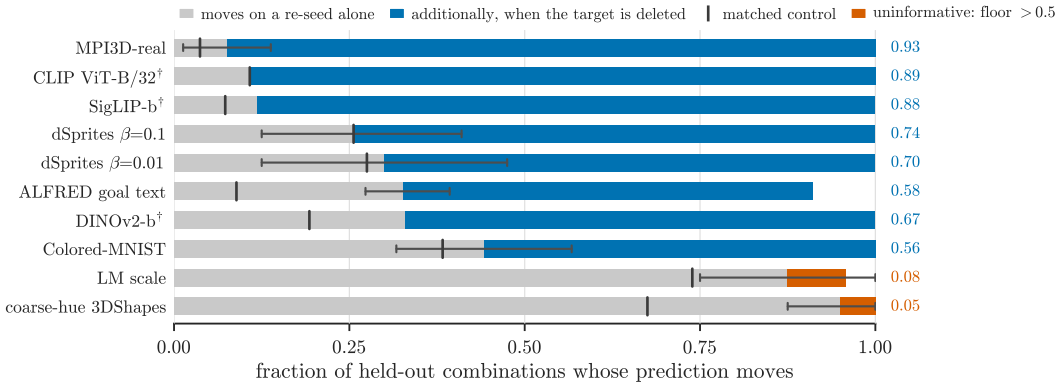

Figure 10: **The re-seed floor decides whether the causal test can be read.** Every row of Table 8 and Table 19, with ALFRED (Appendix B) and language-model scale (Appendix I), on one axis. Each bar is the fraction of held-out combinations whose prediction moves after deleting the captured combination, split at the *re-seed floor*: grey is the share that moves anyway when the split and support are held fixed and only the training seed is redrawn, and the coloured remainder is what the deletion buys on top. **The coloured part is the evidence**. The bar's right-hand end is 0.911 to 1.00 on every substrate, including the two where the test says nothing. Those two are drawn in vermillion: above a floor of 0.5, retraining alone moves the prediction about as often as the intervention does. Ticks mark the matched control deletion and whiskers the seed-clustered 95% interval on the floor. Daggered rows are the frozen encoders, which are exploratory, unregistered and read at the posterior mean, and which retrain a read-out over frozen features rather than from scratch.

We pre-registered six predictions about capture and report all six, including the two that failed. Unless stated otherwise the numbers are dSprites, plain arm, restricted to cells whose held-out composition is zero ($n = 118$ cell$\times$rate rows); the raw records and a written verdict for each prediction ship with the code release.

Read-out choice (pre-registered robustness check). The analysis in this appendix encodes at the posterior mean. Since Section 3 argues stochastic codes should be read by sampling, we re-ran all three findings under a single draw $z \sim q(z|x)$ and an 8-draw average, with thresholds registered in advance. All three tests pass in the sampled read-out: modal mass 0.830 against a 0.50 bar (mean 0.886; 8-draw 0.875), nearest-mean-image predictability 0.761 against a 0.50 bar (mean 0.744), and error-on-seen 0.979 against the 0.824 chance rate. The causal test likewise holds: deleting the target changes the prediction for every cell in every seed, a matched control deletion changes 0.10, and the relocated prediction is the now-nearest remaining seen cell for 0.700 of cells. Sampling raises the predicted-pair entropy ($0.296 \rightarrow 0.475$ nats), as posterior noise should, without dispersing the mass off the modal seen combination. The registered outcome is SURVIVES.

Table 8: **The causal test against a re-seed floor.** Fraction of held-out cells whose prediction moves after each intervention, with seed-clustered 95% CIs. *Paired* counts cells where deletion moved the prediction but re-seeding did not, against the reverse; *re-capture* is the fraction of relocated cells landing on the next-nearest remaining seen combination. The paired difference between deleting the target and re-seeding is $+0.700$ $[+0.525, +0.875]$ on dSprites and $+0.558$ on Colored-MNIST. All four pre-registered tests pass on dSprites at both rates, on Colored-MNIST, and on MPI3D-real (real camera photographs). On 3DShapes three of four fail and the test is uninformative. This protocol supersedes the earlier uncontrolled run, and all dSprites figures of Section 4.3 come from it. Figure 10 plots these rows against the frozen-encoder, ALFRED and language-model-scale floors.

benchmark	delete target	matched control	**re-seed floor**	paired	re-capture
dSprites β=0.01	1.00	0.275	0.300 [0.125, 0.475]	28:0	0.750
dSprites β=0.1	1.00	0.256	0.256 [0.125, 0.410]	29:0	0.564
Colored-MNIST	1.00	0.383	0.442 [0.317, 0.567]	67:0	0.117
MPI3D-real (photographs)	1.00	0.037	**0.075** [0.013, 0.138]	74:0	0.412
coarse-hue 3DShapes	1.00	0.675	**0.950** [0.875, 1.000]	—	0.000

The dissociation band across datasets. Sweeping β for the plain arm on four image settings and classifying each rate by (seen-support accuracy, held-out composition) gives the same ordered three-regime structure everywhere: both decode $\rightarrow$ dissociated (seen fitted, held-out at the floor) $\rightarrow$ joint information loss. The band opens at $\beta \approx 0.1$ on MPI3D colour$\times$shape, 0.3 on dSprites shape$\times$position, and 1 on both Colored-MNIST digit$\times$colour and coarse-hue 3DShapes. We verified the ordering is not a threshold artefact: it holds for every (seen, held-out) cut-off in $\{0.7, 0.75, 0.8, 0.9\} \times \{0.02, 0.05, 0.10, 0.15\}$ that we tried. Across datasets the band exists and the regimes appear in the same order. Only where the band sits varies.

P1 (confirmed): the failure is concentrated. The distribution over predicted combinations for a held-out cell has entropy 0.284 nats against $\log |S| = 2.64$ for a uniform spread over the seen support. One combination takes 0.886 of the mass. Of the error mass, 0.985 lands on *seen* combinations against a chance rate of 0.824, and the target is a seen combination for 0.992 of held cells. The target is cell-specific, with no globally attractive combination: distinct targets across cells 0.983, and the grid centre is the target for only 0.017.

Table 9: P2: which seen combination a held-out cell is captured by, predicted *before* training from the split and the inputs alone. Top-1 accuracy against the model's modal predicted combination. On dSprites the perceptually nearest seen combination is the target, while frozen ImageNet features, sharing a factor value and the highest-degree neighbour are all at chance *on dSprites*. Which control fails is dataset-dependent, with degree tying the pixel predictor on Colored-MNIST and ImageNet features predicting on MPI3D photographs. On the synthetic task, whose inputs are a random nonlinear mixing with no perceptual metric, only the code predicts the target. Seed-clustered 95% intervals are given for the headline mean-image predictor and the control columns are point estimates. This table is the posterior-mean read-out throughout, the only read-out in which all four benchmarks were run, and the sampled dSprites value is 0.761.

benchmark	chance	mean img.	1-NN img.	code	ImageNet	shares fac.	degree	random
dSprites	0.071	**0.745** [0.57, 0.90]	0.339	0.950	0.068	0.054	0.085	0.051
MPI3D (real camera)	0.036	**0.750** [0.55, 0.88]	0.725	0.975	0.675	0.129	0.121	0.000
Colored-MNIST	0.021	**0.183** [0.11, 0.26]	0.142	0.492	0.058	0.069	0.218	0.017
synthetic	0.036	0.155	0.168	0.654	0.155	0.098	0.105	0.015

P2 (confirmed on dSprites, partial on synthetic). Table 9 gives every predictor. On dSprites the seen cell whose *mean image* is nearest is the model's predicted combination 0.745 of the time. This is computed over the 59 failing cells whose captured target lies in the seen support, at the three declared functional rates, rather than the 118 cell$\times$rate rows the concentration statistics pool, because a target outside the seen support has no seen cell to predict. Against a chance rate of 0.071 and a random-scoring control at 0.051; the nearest seen *code* predicts it at 0.950. (Aggregation, for exact regeneration from the released rows: the mean over cell$\times$seed rows at the declared plain operating points $\beta \in \{0.001, 0.01, 0.1\}$ with the target-on-seen-support filter, $n = 59$; per-β means are $0.750/0.800/0.684$ for the image predictor and $0.950/0.900/1.000$ for the code predictor, and their unweighted average reproduces the same two numbers.) Frozen ImageNet ResNet18 features

fail (0.068), and both "shares a factor value" and "highest-degree neighbour" are at chance, so the target is the perceptually nearest seen combination, and sharing a row or column does not predict it. On the synthetic task, where the input is a random nonlinear mixing with no perceptual metric, the target is predictable from the code (0.654) but only weakly from the input space (0.155 against chance 0.036), which is what that generator predicts.

P3 (falsified as stated): capture does not strengthen with compression. We predicted that the ratio $\|\bar{z}_h - z_{\text{nearest seen}}\| / \|\bar{z}_h - z_{\text{additive}}\|$ would fall with the rate penalty. On dSprites the plain code is already fully captured at *every* rate including β=0 (ratio 0.33–0.36, i.e. three times closer to a seen combination than to the additive point) and the trend in β has the wrong sign (Spearman $+0.16$). On the synthetic task the ratio does fall (Spearman -0.392) but never crosses 1 in the operating regime. What separates captured from uncaptured is the *arm* at matched rate: at β=0.01, plain 0.35 against routed 1.42. We therefore describe capture as a property of the shared read-out, which compression exposes and does not create.

P4 (confirmed): routing is captured per block, which is what makes it composition. For held-out inputs, per-factor nearest-prototype accuracy on dSprites at β=0.01 is 0.17 in the plain arm, at or below the 0.25 chance rate, against 0.63 routed, with held-out composition 0.00 against 0.25. On the synthetic task at $\beta \in \{0.5, 0.7\}$: plain $0.63/0.46$ against routed $0.98/0.96$, with composition $0.28/0.07$ against $0.97/0.93$. The nearest seen cell *within* a block shares that factor's value $0.74/0.47$ routed against $0.104/0.201$ plain. Being captured by the nearest value of each factor separately is correct composition; being captured by the nearest whole combination is not.

P5 (falsified, sign reversed): distance from the support predicts success. We predicted that cells far from the seen support would fail most. On the dSprites plain arm, scored with the graded outcome (mean marginal head accuracy, since joint accuracy is at the floor for every cell), $\rho = +0.44$ for the pixel-space support gap and $+0.56$ for the code-space gap: cells far from every seen combination do *better*. This is coherent with capture: what hurts a cell is having a close seen neighbour available to be captured by, but it is the opposite of what we wrote down, and we report it as a failed prediction. The support-*geometry* statistic survives on the same cells (leverage -0.36, per-value degree control -0.44).

Table 10: P6, the causal test. At an operating point where held-out composition is zero, delete the capture target from the seen support and retrain; the control deletes an equally sized seen combination that is nobody's target. "changed" is the fraction of held-out cells whose predicted combination moved. dSprites rows are 10 seeds with seed-cluster 95% CIs.

benchmark	condition	n	changed	new target is pixel-nearest remaining
dSprites (β=0.01)	delete target	40	**1.00** [1.00, 1.00]	0.750 [0.60, 0.875] (chance 0.08)
dSprites (β=0.01)	delete control	40	0.275 [0.175, 0.375]	—
dSprites (β=0.1)	delete target	39	**1.00** [1.00, 1.00]	0.564 [0.40, 0.744] (chance 0.08)
dSprites (β=0.1)	delete control	39	0.256 [0.125, 0.410]	—
synthetic (β=1.0)	delete target	40	**1.00**	—
synthetic (β=1.0)	delete control	40	0.55	—

P6 (confirmed, causal). Table 10 gives the full intervention. Deleting the capture target from the training support and retraining relocates the held-out prediction in every dSprites cell across 10 seeds at β=0.01 (n=40; seed-cluster CI [1.00, 1.00]), against 0.275 [0.175, 0.375] when a matched control combination is deleted, and the new prediction is again the pixel-nearest *remaining* seen combination 0.750 [0.60, 0.875] of the time against a chance rate of 0.08. The same tests pass at a second rate (β=0.1: target 1.00, control 0.256 [0.125, 0.410], re-capture 0.564 [0.40, 0.744]). This rules out a fixed perceptual confusion: the prediction tracks what the support contains. On the synthetic task the same contrast is 1.00 against 0.55; that rate sits at the edge of collapse, which makes the control noisy, and we report the dSprites contrast as the clean one.

P6b (pre-registered, confirmed): the relocation is a cascade. Table 12 iterates the deletion. For every captured held cell of the published β=0.01 split, seeds 0–4: delete the observed target t_1, retrain from scratch, read the new target t_2; delete $\{t_1, t_2\}$, retrain, read t_3; delete $\{t_1, t_2, t_3\}$,

retrain, read t_4. The prediction at each depth is compared with the seen cell whose mean image is nearest to the held cell *among the cells that remain*, a ranking computed from the image bank alone. Controls under the same trainer and step count: delete $k \in \{1, 2, 3\}$ seen cells that are nobody's target (*random-k*), and delete nothing but redraw the training seed (*re-seed*). All fourteen pre-registered tests pass. The rank distribution of the new target over the 54 chain steps is 0.70 at rank 1, 0.13 at rank 2, 0.09 at rank 3 and 0.07 beyond; 5 of the 17 full-depth chains are nearest at all three depths, against $\sim 5 \times 10^{-4}$ under chance. In the control arms the prediction mostly does not move, so their "nearest" rates are inherited from the original target and are not the comparison; the comparison is the relocation rate. Two chains ended early because the relocated prediction was no longer a captured seen cell, which is why n falls from 19 to 17.

P6c (pre-registered): the cascade on real photographs. The same procedure, schedule and bookkeeping, with the MPI3D-real adapter of the predictability run substituted (real camera photographs, object-colour×object-shape, 8 held cells per seed, β=0.1, the single rate registered for the causal test there), seeds 0–4, 40 chains, 116 chain steps, pre-registered. Table 11: the prediction relocates at every step of every chain; the new target is the pixel-nearest remaining cell in 0.43, 0.42 and 0.53 at depths one to three against 0.04 chance, within the top two in 0.75, mean rank 2.6 against 13.5; random-k deletion relocates 0.05–0.075 and re-seeding 0.075. The rank distribution is 0.46, 0.29, 0.06 at ranks 1–3 and 0.10 beyond; 7 of 38 full-depth chains are nearest at all three depths (6×10^{-5} under chance). Thirteen of the fourteen tests pass; the failing one is the pooled top-one bar of 0.5 (observed 0.457 $[0.375, 0.573]$), which the pre-registration flagged as calibrated on dSprites and which matches the one-step re-capture rate of 0.412 on this benchmark. Seen-support accuracy after every deletion is 0.978.

Table 11: Multi-step deletion cascade on real camera photographs (MPI3D-real, plain arm, β=0.1, seeds 0–4). Columns as in Table 12; chance is $1/n_{\text{remaining}}$ with 27, 26, 25 remaining cells.

arm	depth	n	relocates	new target is nearest	top-3	mean rank	chance (top-1; rank)
cascade	k=1	40	1.00 [1.00, 1.00]	0.425 [0.28, 0.55]	0.78	2.90 [2.10, 3.70]	0.037; 14.0
cascade	k=2	38	1.00 [1.00, 1.00]	0.421 [0.32, 0.57]	0.79	2.42 [2.00, 2.92]	0.038; 13.5
cascade	k=3	38	1.00 [1.00, 1.00]	0.526 [0.46, 0.62]	0.87	2.39 [1.82, 3.05]	0.040; 13.0
cascade	pooled	116	1.00 [1.00, 1.00]	**0.457** [0.38, 0.57]	0.81	2.58 [2.12, 3.00]	0.038; 13.5
random-k	k=1	40	0.075 [0.00, 0.18]	—	—	—	—
random-k	k=2	40	0.075 [0.00, 0.18]	—	—	—	—
random-k	k=3	40	0.050 [0.00, 0.10]	—	—	—	—
re-seed	—	40	0.075 [0.00, 0.18]	—	—	—	—

Table 12: Multi-step deletion cascade on dSprites (plain arm, β=0.01, seeds 0–4). "Nearest" is the pixel-nearest seen cell among those remaining after the deletion, fixed before training; chance is $1/n_{\text{remaining}}$. Seed-cluster bootstrap 95% CIs.

arm	depth	n	relocates	new target is nearest	top-3	mean rank	chance (top-1; rank)
cascade	k=1	19	1.00 [1.00, 1.00]	0.737 [0.50, 0.95]	1.00	1.42 [1.10, 1.78]	0.077; 7.0
cascade	k=2	18	1.00 [1.00, 1.00]	0.556 [0.39, 0.71]	0.83	2.33 [1.50, 3.42]	0.083; 6.5
cascade	k=3	17	1.00 [1.00, 1.00]	0.824 [0.50, 1.00]	0.94	1.35 [1.00, 2.00]	0.091; 6.0
cascade	pooled	54	1.00 [1.00, 1.00]	**0.704** [0.59, 0.82]	0.93	1.70 [1.31, 2.22]	0.083; 6.5
random-k	k=1	20	0.15 [0.05, 0.25]	—	—	—	—
random-k	k=2	20	0.30 [0.10, 0.55]	—	—	—	—
random-k	k=3	20	0.15 [0.00, 0.35]	—	—	—	—
re-seed	—	20	0.35 [0.15, 0.50]	—	—	—	—

P6d (pre-registered): matched-distance deletion. The remaining rival to "the read-out selected this cell" is "any deletion *near* the held cell perturbs the local support and relocates the prediction". Pre-registered. dSprites shape×scale, plain arm, β=0.01, seeds 0–9, the protocol of Table 8. For each of the 40 captured cells we delete, instead of the target, the non-target seen cell whose pixel distance to the held cell is closest to the target's, and retrain. That cell is the pixel second-nearest in $31/40$ cells and sits at $1.40\times$ the target's distance on average ($1.07\times$ in the 8 cells where the second-nearest itself was deleted), so the registered "within $1.25\times$" gate is not met: the support is discrete and no cell sits at the target's distance. Deleting it relocates the prediction in 0.200 $[0.075, 0.350]$ of cells (0.375 $[0.11, 0.80]$ in the eight second-nearest deletions), against 1.000 for deleting the target, 0.275

for a random non-target and 0.300 for re-seeding; seen accuracy after retraining is 0.98. Relocation follows the selected cell itself.

Why extra information cannot help. Within every one of these runs the same frozen heads read two things: a real held-out image, and the composed code built from per-value prototypes. On dSprites at $\beta{=}0.01$ the composed code reads out at 0.85 while the real held-out image reads out at 0.00 through the identical heads. The additive structure *is* in the code and the heads *can* use it; the encoder simply never places a held-out image at the additive point. This is what makes capture a placement failure with the information intact, and it is why the on-support additivity penalty of Appendix J cannot reach it.

E.1 Alternative explanations, each with the test that decides it

Table 13 lists the rival accounts of the capture data a reviewer might raise, what each predicts, the test in this paper that bears on it, and the outcome. Two are excluded by construction: cell frequency and class balance, because every trainer draws seen cells *uniformly*, so no seen combination is more frequent than another and the target cannot be the majority class. The distance-metric row is new: re-scoring the 59 published predictability rows with cosine, per-pixel standardised and correlation distances on the same mean images gives top-one rates of 0.373, 0.678 and 0.322 against 0.746 Euclidean (chance 0.071), so the prediction survives the metric, with the scale-invariant metrics losing exactly the scale factor's ordering.

Table 13: Alternative explanations of the capture data and the test that decides each. dSprites shape×scale plain arm unless stated; numbers are those reported in Section 4, Section 5 and this appendix.

alternative	it predicts	test	outcome
Information loss: the code no longer carries the factors	composition and seen decodability fall together; no read-out recovers held-out cells	prototype-composition operator through the *same* heads; Fano certificate; held-out-conditioned probe	operator 0.858 vs. real input 0.020; deficit 0.235 of 3.584 nats; MPI3D-real code retains $2.06/2.94$ nats. Not sufficient as a cause
Fixed perceptual confusion, support-blind	deleting the target leaves the prediction where it was	delete target and retrain; re-seed and control arms	relocates 1.00 vs. re-seed 0.300, control 0.275; paired $28{:}0$
Local support perturbation: any nearby deletion relocates	deleting an equally close non-target relocates as often	delete the nearest non-target cell instead (P6d)	relocates 0.200 $[0.075, 0.350]$, at the re-seed floor, vs. 1.00 for the target
Two independent per-factor value biases	deleting one *combination* whose values both remain moves nothing; relocation lands on a shared-value cell	restrict to such deletions; shared-value null	$37/40$ still relocate, paired $25{:}0$ against re-seeding and $26{:}0$ against the matched control at $\beta{=}0.01$, repeating at $\beta{=}0.1$; re-capture $3.7\times$ ($\beta{=}0.01$) and $2.7\times$ ($\beta{=}0.1$) the shared-value null, $p < 10^{-6}$ both; product of marginals fixes the *amount* (0.874 of 0.886) and leaves the address open
Pretrained perceptual similarity	frozen ImageNet features name the target	ResNet18 feature-nearest seen cell	0.068 on dSprites (chance); predicts on MPI3D photographs, as expected of photographs
Hub cell / support degree	the most connected seen cell is the target	highest-degree neighbour; most central cell	0.085; central cell is the target 1.7% of cells; targets distinct in 98.3%
Seed variance	retraining alone moves the prediction as much	re-seed arm, 10 seeds, all benchmarks	floor 0.300 dSprites, 0.075 MPI3D-real; deletion 1.00 on both
Cell frequency or class imbalance	the majority seen cell captures	excluded by design: uniform cell sampling in every trainer	—
Mean read-out artefact	capture disappears under a single draw	pre-registered sampled re-run of all three findings	all three survive
Head artefact	capture lives in the softmax heads alone	nearest-code analysis without heads; rate-grid pooling	present in code geometry alone; robust to pooling
Euclidean metric artefact	the pixel predictor works only under one distance	cosine, standardised, correlation on the same means	$0.373, 0.678, 0.322$ vs. 0.746; all $\geq 4.5\times$ chance
Per-factor rate allocation	a protected per-group rate restores composition	per-group vs. global KL floor, 1148 models	$+0.014$, indistinguishable from global; hurts when added to routing
Reduced fan-in	fewer dimensions per head is what helps	both heads read the *same* half	below plain; 99.9% of rate inside the read dimensions
Post-hoc geometry: the predictor is fit to the data	on geometries fixed in advance the rule fails	controlled ordinal geometry, target and margin set before training, 838 models (Appendix E.2)	names the pre-specified target in 0.341 $[0.303, 0.367]$, $8.6\times$ chance, above the shares-a-value null; below the registered 0.5 bar and below dSprites' 0.745

E.2 CAPTURE ON GEOMETRIES FIXED BEFORE TRAINING

On the dSprites capture rows the trained code preserves the pixel-space ordering of seen cells in 0.653 $[0.548, 0.758]$ of cells, a partial bridge covering two thirds of cells.

The predictability result of Section 4.2 measures a rule on data it did not choose: the seen support is whatever the random split gave, and the pixel metric is read off images that already exist. A reviewer may reasonably ask whether the rule is fitted to those geometries. This experiment removes that freedom by *constructing* the geometry: a two-factor ordinal grid in which the spacing ratio r between the two factors' value spacings and a gap γ inserted into one factor's ladder are set by us, so the intended captured cell $s^\star(u) = \arg\min_{s \in S} d(u, s)$ and its margin over the runner-up are both known *before any model is trained*. Pre-registered. Design: $r \in \{1, 1.5, 2, 3\}$, $\gamma \in \{0, 0.5\}$, 4 and 12 held-out cells, $\beta \in \{0.01, 0.1, 0.3, 1\}$, arms plain / 50%-overlap / routed, seeds 0–5, 838 models and 6600 cell rows. The γ=0 sweep was stopped after seed 5, so the pooled analysis uses the completed seeds; every rate below is over those.

The rule predicts, but not at the strength it does on dSprites. Over 1104 held-out cells of the plain arm inside the window, 1027 are captured (0.930 land on a seen cell) and the *pre-specified* target is the one captured in 0.341 $[0.303, 0.367]$ of them, against a 0.040 chance rate: $8.6\times$ chance, top-two 0.557, mean rank 4.3 of 13.2. It also clears the shares-a-factor-value null (0.553 observed against 0.139 expected). So the rule is *not* fitted to the geometries it was read off: on geometries chosen in advance it still names the captured cell far above chance. But the registered bar was 0.5 and it is not met, and 0.341 is well below the 0.745 the same rule reaches on dSprites. 4 of 7 tests pass.

What the dial does and does not do. Widening the spacing ratio raises predictability monotonically, $0.305, 0.310, 0.336, 0.412$ at $r = 1, 1.5, 2, 3$ (registered G2b, passes), and the *direction* of the misread follows the design: the fraction of captures that keep factor b and misread a rises from 0.19 to 0.47 as r grows, which is what widening a's spacing should do. The continuous margin, however, does not grade the outcome within a geometry: Spearman 0.10 $[-0.06, 0.25]$, so registered G2 fails. The dial works between designs and not within them.

Specialisation reduces capture; overlap does not separate from plain. At the plain arm's window points, capture rate is 0.930 plain, 0.903 at 50% overlap and 0.842 routed; routed is below overlap by 0.061 $[0.029, 0.099]$ (registered G3b, passes) while plain over overlap is 0.027 $[-0.007, 0.068]$ and does not clear zero (G3 fails). Held-out composition moves the other way, routed above plain by 0.072 $[0.031, 0.129]$. The ordering of Section 5 therefore reproduces on geometry we built, with the plain-versus-half-overlap step too small to resolve here.

The encoder does not inherit the geometry it was given. The most informative negative: the cell nearest in the *trained latent* predicts the captured cell at 0.769, more than twice the 0.341 of the designed target, and the latent-nearest and design-nearest cells agree only 0.397 of the time. Fixing the input geometry does not fix the code geometry; the encoder builds its own metric, and that metric is what capture follows. This is the same boundary the main text draws around Proposition 1, selection is modelled, *placement* is not, now measured directly rather than inferred, and it is why a designed-geometry predictor cannot be expected to match a pixel predictor read off the images the encoder actually sees.

F FOUR-ARM COMPARISON

This is the comparison that motivated the factorial of Appendix G. Three ways of giving each factor its own machinery, routed blocks of one code, fully independent per-factor bottlenecks, and independent bottlenecks over a shared trunk, all beat the plain shared token at matched achieved total rate, and none separates from the others (Table 14). Because all three confound a protected per-factor rate with a specialised read-out, the comparison cannot say which ingredient matters; that is exactly what Section 5 decomposes.

Table 14: Four arms at matched achieved total rate (synthetic, 5 seeds, chance 0.028; all arms reach seen accuracy 1.00 throughout). Every factored arm beats the plain shared token and they do not separate from each other. Read with Section 5, which decomposes what they share: the common ingredient that reduces capture is factor-specialised read-out subspaces. The per-factor rate allocation these arms also happen to share does not reduce it on its own.

achieved rate (nats)	plain VIB	routed blocks	independent	indep., shared trunk
2.6	0.14 ± 0.06	$\mathbf{0.93 \pm 0.07}$	0.97 ± 0.05	0.87 ± 0.05
3.3	0.45 ± 0.09	$\mathbf{0.94 \pm 0.07}$	0.97 ± 0.04	0.94 ± 0.02
4.3	0.74 ± 0.13	$\mathbf{0.94 \pm 0.07}$	0.98 ± 0.05	1.00 ± 0.01
5.7	0.88 ± 0.07	$\mathbf{0.95 \pm 0.06}$	0.99 ± 0.02	1.00 ± 0.00
7.4	0.96 ± 0.05	0.96 ± 0.06	0.98 ± 0.03	1.00 ± 0.00
12.4	0.98 ± 0.04	0.99 ± 0.03	1.00 ± 0.00	0.99 ± 0.03
mean	0.69	**0.95**	0.98	0.96

G MECHANISM ISOLATION: THE FULL FACTORIAL

The full synthetic factorial behind Section 5, all 15 arms under both read-outs, with the instruments that measure specialisation.

The information-floor arm. It penalises the shortfall of $I(Z_g; y_f)$ below a target and leaves the read-out unrestricted. Its sampled gain over the plain code is 31% of the reference routing gap of 0.328, and 33% of the routing gain measured within its own run.

Table 15: **Learned and installed specialisation, as methods.** Sampled read-out at matched achieved rate unless a mean figure is named. "Gap" is held-out composition above the plain shared token. AIN is the attribute-invariance objective. Group sparsity is given no partition and no factor-to-dimension map. The two frozen-3DShapes rows read different blocks of the same run and are not pooled: the group-sparsity row ($+0.212$ vs. routing $+0.209$) is the factor-count block, and the AIN row ($+0.302$, with group sparsity $+0.240$ and routing $+0.228$) is the main block, where the plain baseline differs.

setting	arm	gap	routing	note
synthetic	group sparsity (learned)	$+0.350$	$+0.328$	full recovery
synthetic	alignment, no floors	—	—	46%, mean 74%
synthetic	alignment − floors	$+0.137$	—	alignment wins
frozen 3DShapes	group sparsity	$+0.212$ $[+0.192, +0.231]$	$+0.209$	ties
frozen 3DShapes	AIN	$+0.302$	$+0.228$	AIN wins
UT-Zappos50K	group sparsity	—	—	92% of routing
UT-Zappos50K	AIN − group sparsity	$+0.004$ $[-0.005, +0.013]$	—	tie

Design. The sweep behind Figure 3 and Table 16 is a factorial over read-out geometry × rate knob × latent width × seed on the synthetic 6×6 task, using the generator and protocol of Appendix B unchanged. Fifteen arms × 10 seeds × a rate sweep give 1100 trained models at the paper's width $D = 16$, with 290 more at $D = 8$, 290 at $D = 32$ and 96 on dSprites: 1776 models in total. Every comparison is made at *matched achieved rate*, never at matched penalty weight: each arm is swept over its rate knob, each seed's rate–composition curve is interpolated in log-rate onto a common grid, and arms are compared in the crux band 1.5–5.0 nats where the plain code's composition is actually falling. Intervals are 95% paired t intervals over the 10 seeds.

Arms. *plain shared*: both heads read all Z dimensions. *routed blocks*: head f reads its own disjoint block of $Z/2$. *routed + decorrelation*: adds the cross-block penalty. *overlap* ρ: both heads have fan-in $Z/2$ and share a ρ fraction of their dimensions, so ρ=0 is routing and ρ=1 is two heads on the *same* random half, fan-in reduced, disjointness zero. *fan-in* 12, 8 *shared*: the literal $[0{:}12]/[4{:}16]$ version. *rate floor per latent group / per dimension / global*: a plain full-fan-in read-out with the KL floored per contiguous $Z/2$ group, per individual dimension, or on the total only, rate protection with no read-out structure at all. *aux. alignment*: a plain full-fan-in read-out over a code pushed into factor-aligned subspaces by an alignment objective, no floors. *group sparsity*: a group-sparsity penalty on the read-out weights that must *discover* where to specialise, with an unrestricted architectural mask.

Table 16: Every arm, both read-out currencies, at matched achieved rate (10 seeds, crux band 1.5–5.0 nats). Δ is paired against the plain shared code with a 95% interval. All arms fit the seen support at 0.99–1.00 except group sparsity, which reaches 0.93 in this band. One design caveat: group sparsity has a single measured operating point inside the crux band (achieved rate 4.08 nats, neighbours at 0.49 and 6.50), so its crux-band figures are largely log-interpolated, unlike the other arms which have two or more. Rate protection with an unstructured read-out (*rate floor per latent group*) does not recover and is indistinguishable from a purely global floor; reduced fan-in without disjointness (*overlap* 1.00) sits *below* plain; what recovers is factor-specialised read-out subspaces, whether hard-wired, installed by an auxiliary objective, or learned.

	mean read-out		sampled read-out	
arm	held-out	Δ vs. plain	held-out	Δ vs. plain
plain shared	0.335	+0.000	0.129	+0.000
routed blocks	0.915	+0.580 [+0.467, +0.693]	0.457	+0.328 [+0.270, +0.386]
routed + decorrelation	0.902	+0.567 [+0.477, +0.658]	0.514	+0.385 [+0.336, +0.434]
group sparsity (learned)	0.888	+0.553 [+0.461, +0.645]	0.479	+0.350 [+0.280, +0.421]
aux. alignment, no floor	0.763	+0.428 [+0.312, +0.544]	0.279	+0.151 [+0.110, +0.192]
routing + rate floors	0.825	+0.490 [+0.381, +0.600]	0.381	+0.252 [+0.216, +0.289]
group floor + alignment	0.654	+0.319 [+0.170, +0.468]	0.228	+0.100 [+0.055, +0.145]
rate floor per dimension	0.571	+0.236 [+0.124, +0.348]	0.236	+0.108 [+0.060, +0.155]
rate floor per latent group	0.309	−0.026 [−0.149, +0.097]	0.142	+0.014 [−0.029, +0.056]
global total-rate floor	0.304	−0.031 [−0.116, +0.054]	0.132	+0.004 [−0.038, +0.045]
overlap 0.25	0.515	+0.180 [+0.063, +0.297]	0.217	+0.089 [+0.036, +0.141]
overlap 0.50	0.400	+0.065 [−0.006, +0.136]	0.160	+0.031 [−0.007, +0.069]
overlap 0.75	0.362	+0.027 [−0.075, +0.129]	0.121	−0.007 [−0.031, +0.017]
overlap 1.00 (same dims)	0.259	−0.076 [−0.156, +0.005]	0.080	−0.049 [−0.074, −0.023]
fan-in 12, 8 shared	0.283	−0.052 [−0.189, +0.084]	0.111	−0.017 [−0.056, +0.021]

Two floors that do move, and why they do not rescue the rate account. Two rate-side arms in Table 16 are not null. A *per-dimension* floor recovers part of the gap (+0.108 sampled), and *routing plus floors* stays well above plain (+0.252). Neither supports rate allocation as the mechanism. The per-dimension floor forces every latent coordinate to stay active, which is a diffuse participation constraint rather than a factorwise one; it is the *group* floor, the arm that actually implements "each factor gets its own protected share", that is null and indistinguishable from a global floor. And routing-plus-floors is worse than routing alone (−0.090 mean, −0.076 sampled at matched rate): adding rate protection to an arm that already works costs composition. The same holds for floors added to a factor-aligned code (−0.051). Read together, the two arms that move are the ones that do not isolate factorwise rate, and every arm that does isolate it is null or negative. (The released results mark factorwise rate as helping in the mean currency only because they pool in the floor+alignment arm (+0.319 mean), whose mover is the alignment, as the decomposition above shows. The pure per-factor floor recovers in neither currency.)

Table 17: The overlap dependence across latent widths. Composition against subspace overlap $0.00, 0.25, 0.50, 0.75, 1.00$ at fixed fan-in $Z/2$, with the per-seed slope and its 95% interval. The dependence is graded and monotone at $D = 8$ and $D = 16$; at $D = 32$ it is abrupt, with any overlap collapsing composition outright.

width	currency	composition at overlap 0.00 → 1.00	per-seed slope	monotone
$D = 8$	mean	0.845, 0.666, 0.551, 0.274, 0.042	−0.799 [−0.884, −0.713]	yes
$D = 8$	sampled	0.348, 0.272, 0.230, 0.107, 0.025	−0.324 [−0.358, −0.290]	yes
$D = 16$	mean	0.915, 0.515, 0.400, 0.362, 0.259	−0.586 [−0.654, −0.518]	yes
$D = 16$	sampled	0.457, 0.217, 0.160, 0.121, 0.080	−0.340 [−0.382, −0.298]	yes
$D = 32$	mean	0.849, 0.229, 0.157, 0.167, 0.231	−0.519 [−0.681, −0.358]	no
$D = 32$	sampled	0.319, 0.085, 0.049, 0.053, 0.057	−0.222 [−0.250, −0.194]	no

The positive diagnostic. A crossed per-group factor probe reads how much each latent group carries the *other* factor; it is low when each group carries only its own. Across the fifteen arms it correlates $r = -0.731$ with crux-band composition, against $r = +0.595$ for rate balance and $r = +0.390$ for participation (mean read-out; sampled: −0.637, +0.517, +0.327, the same ordering, weaker). Arm

by arm, over the full factorial population (which is wider than the information-floor comparison quoted in Section 5, hence the slightly higher values), the probe reads 0.358 for routing, 0.354 for routing with floors and 0.454 with decorrelation, against 0.861 plain, 0.915 for the group-floored arm and 0.999 for the per-dimension-floored one. A KL floor on a block of dimensions does not make that block carry a particular factor, which is the whole reason rate protection cannot substitute for specialisation. The plain code's achieved rate is already 0.761-balanced across its two halves, so it is not failing from lopsided allocation in the first place.

Joint regression at matched rate. The arm-level correlations above enter one predictor at a time. Entering them together, model by model, gives partial effects: over the 329 functional models (seen ≥ 0.9) inside the crux band, standardised coefficients of sampled composition on architectural overlap, log achieved rate, the crossed probe and rate balance are -0.289 $[-0.362, -0.231]$, $+0.634$ $[+0.608, +0.667]$, -0.283 $[-0.346, -0.227]$ and $+0.191$ $[+0.118, +0.245]$ (seed-cluster bootstrap, $R^2 = 0.69$; posterior mean: -0.143, $+0.535$, -0.387, $+0.261$). Three readings. Rate is *not* irrelevant: inside a $3\times$ band of achieved rates, more bits still buy composition at fixed geometry, which is the necessary-condition half of Section 3. At *fixed* rate, overlap and cross-factor leakage each lower composition independently, with intervals well clear of zero. And overlap creates leakage: regressing the crossed probe on overlap and rate gives $+0.446$ $[+0.410, +0.486]$ for overlap, so part of overlap's effect runs through the probe and part does not, the "partly mediated" pattern group sparsity predicts (it lowers *learned* overlap while the probe of *pre-specified* groups cannot see it). Pooled over the whole sweep (0.004–36 nats, 912 models) log rate alone reaches $+0.84$ and the probe's partial effect vanishes, which is why the paper compares arms at matched achieved rate in the first place: across a 10^4 range of budgets, bits explain composition trivially.

Learned specialisation. The group-sparsity arm reaches head supports whose learned weight overlap is 0.000 at every functional operating point, despite an unrestricted architectural mask, and its composition is statistically indistinguishable from hard-wired routing (sampled $+0.350$ $[+0.280, +0.421]$ against routing's $+0.328$ $[+0.270, +0.386]$). Specialisation therefore does not have to be supplied by the architect; it has to be present.

Group sparsity as a method. The same objective, now compared at matched achieved rate against an attribute-invariant baseline and three disentanglement objectives on DINOv2 embeddings, recovers the routed gap on 3DShapes object-hue $\times$ scale ($+0.212$ $[+0.192, +0.231]$ vs. routing $+0.209$, 10 seeds, sampled) and 92% of it on UT-Zappos50K material $\times$ type ($+0.048$ $[+0.040, +0.056]$; routing $+0.052$; AIN $+0.044$; β-VAE / FactorVAE / β-TCVAE at or below plain). A seen-accuracy-matched read leaves the ranking unchanged. When the factor *count* is misspecified, $F'{=}2$ (the true count) is the peak; merging into one class yields zero held-out composition, and over-decomposition ($F'{=}3, 4$) keeps only about half the gap (code release).

Regenerating these numbers. Every number here is regenerable from the code release without retraining. The dSprites replication in that release is a 3-seed, 8-arm run that is underpowered and null on its own; we do not include it in any claim above.

H THE MECHANISM FACTORIAL ON FROZEN FOUNDATION ENCODERS

The factorial of Appendix G re-run on frozen foundation encoders, the replication stated in Section 5.

Why this run exists. The factorial of Appendix G is 1100 models on one synthetic generator, and its small dSprites companion is underpowered and null. That is the objection we would raise ourselves, so we re-ran the whole design where each model is a small head over *cached frozen features* and is therefore cheap enough to run at scale.

Protocol. The code imports the arm definitions, read-out masks, rate penalties, group-sparsity penalty, probes and the entire matched-rate analysis *verbatim* from the synthetic implementation; only the data adapter differs (an MLP head over cached embeddings in place of the synthetic generator). 4280 models: four encoder$\times$composition cells $\times$ 107 models per seed $\times$ 10 seeds, at latent width $D = 16$, with the same per-seed split rng (the per-seed grid is 8 β-swept arms at 10 rate knobs

Table 18: The mechanism factorial re-run on frozen foundation encoders (3DShapes (Burgess & Kim, 2018), 10 seeds; MPI3D (Gondal et al., 2019), 6 seeds, registered before the run, sampled read-out, matched achieved rate, crux band 2–5 nats). Γ is the routed-minus-plain gap each arm must explain. *Floor gap* is per-factor floor minus a single *global* floor: if rate allocation were the mechanism this would be a large positive number, and it is not. All five rows read the *core-arm* blocks of their respective artefacts (the arm set common to every cell), so the rows are mutually comparable. The synthetic reference row is the same factorial read in *its own* pre-registered crux band (1.5–5.0 nats) rather than the frozen cells' 2–5 nats, so it is a reference point and not a band-matched comparison.

frozen encoder, factors	Γ	fan-in only	floor gap	overlap slope	learned sparsity
DINOv2-base, hue×scale	0.226	$+0.004$	$+0.001$ $[-0.003, +0.005]$	-0.219 $[-0.242, -0.197]$	88%
SigLIP-base, hue×scale	0.226	$+0.002$	-0.000 $[-0.002, +0.002]$	-0.220 $[-0.232, -0.207]$	72%
CLIP ViT-B/32, hue×scale	0.203	-0.003	-0.001 $[-0.002, -0.000]$	-0.199 $[-0.214, -0.184]$	83%
CLIP ViT-B/32, hue×shape	0.218	-0.036	$+0.011$ $[+0.004, +0.018]$	-0.244 $[-0.272, -0.217]$	84%
DINOv2-base, MPI3D col.×shape	0.220	-0.009	$+0.000$ $[-0.008, +0.009]$	-0.223 $[-0.268, -0.179]$	98%
synthetic (reference)	0.328	-0.049	$+0.010$ $[-0.024, +0.044]$	-0.340 $[-0.382, -0.298]$	107%

plus 3 floor arms at 9). Cells are DINOv2-base, SigLIP-base and CLIP ViT-B/32 on 3DShapes object-hue×scale, plus CLIP ViT-B/32 on object-hue×shape as a second composition. A fifth cell on a *second dataset*, DINOv2-base on MPI3D object-colour×object-shape (6×6 grid, 9 held cells, 6 seeds, 642 further models), was pre-registered and is reported below whichever way it came out. Arms are compared at matched *achieved* rate in a pre-registered crux band of 2–5 nats, taken from the operating range of the frozen-encoder frontier already reported in Appendix K, never at matched β.

What replicates. Table 18 gives the headline, now including the registered MPI3D cell: there every ordering reproduces, routed gap $+0.220$, learned group sparsity recovering 98% of it (learned head-weight overlap 7×10^{-4}), per-factor vs. global floor $+0.000$ $[-0.008, +0.009]$ (no separation), fan-in only -0.009, overlap slope -0.223 $[-0.268, -0.179]$, negative in every seed though with a terminal uptick ($0.064 \to 0.068$), so not monotone in this cell. All four questions the synthetic sweep answers are answered the same way here: rate protection with an unstructured read-out does not recover and does not separate from a single global floor; reduced fan-in without disjointness does not recover; the overlap slope is negative in every cell and seed, monotone in all readings except two trivial terminal upticks (CLIP hue×shape, sampled, $0.049 \to 0.051$; MPI3D, $0.064 \to 0.068$), and both read-out currencies; and specialisation that must be *learned* recovers most of the gap with a learned head overlap at or near zero. The arm-level verdict flags reproduce exactly: (a) protected per-factor rate FALSE, (b) reduced fan-in FALSE, (c) disjointness TRUE.

A refinement the synthetic task did not show. Across arms the *crossed* factor probe predicts composition at $r = -0.70$ to -0.80 on the four 3DShapes cells, as it does on the synthetic task, and at $r = -0.61$ on MPI3D, weaker, same sign, still the dominant predictor there. The *aligned* probe, how well a block carries its *own* factor, is near zero on three 3DShapes cells (-0.078, $+0.034$, $+0.049$) and on MPI3D (-0.09), against $+0.53$ synthetically; the CLIP hue×shape cell is the exception at $+0.211$. On real features, what predicts composition is mostly the *absence* of the other factor from a block, and the presence of its own factor matters less. Rate balance is the predictor whose sign is not stable: -0.641, -0.560, -0.334 and $+0.037$ across the four 3DShapes cells and $+0.281$ on MPI3D. We therefore read the crossed probe as the portable predictor and treat rate balance as uninformative, with no reliable sign.

Four caveats we would rather state. (i) The one partial non-replication: on the CLIP hue×shape cell a per-factor floor does beat a global floor, $+0.011$ $[+0.004, +0.018]$ sampled ($+0.042$ mean). It is 5% (sampled) or 11% (mean) of the gap to explain and appears in none of the other three cells, and we report it. (ii) In the crux band the floor arms sit $+0.002$ to $+0.025$ above plain, 1–12% of Γ, rather than exactly at it; in the *functional* band where every arm fits the seen support at ≥ 0.90 this reverses and all three floor arms fall *below* plain (-0.003 to -0.067) while routed and group-sparse stay far above. Both bands are in the released tables. (iii) The group-sparsity arm fits the seen support worse in the crux band (0.61–0.95 against 0.84–1.00), which handicaps it, and it still wins; its advantage is larger in the functional band. (iv) Effect sizes are smaller than on the synthetic task ($G \approx 0.21$ against 0.32), as expected when the representation is frozen and not trained for the task.

Relation to a known negative. An earlier attempt in this project found that a pretrained ImageNet ResNet18 backbone *destroyed* composition. That was the shape×scale pairing, where the frozen features do not carry the factors across held-out combinations at all. The runs here use object-hue×scale, the composition for which Appendix K already reports a large routed-versus-plain separation, so the two results are consistent: read-out geometry can only matter where the factors survive in the features. Full records, matched-rate curves, per-cell contrasts and written verdicts ship with the code release.

Capture itself on the same three encoders. The factorial above is the *mechanism* on frozen encoders. We also ran the three capture findings of Section 4 on the same substrate, 3DShapes object-hue × scale over cached DINOv2-b, SigLIP-b and CLIP ViT-B/32 features, a plain shared bottleneck with $|S|{=}60$ of 80 combinations and 20 held out. Deleting a combination retrains the bottleneck and the heads over the *frozen* cached features, so this is a read-out-level intervention and not the retrain-from-scratch of Table 8.

Table 19: **The causal test on three frozen foundation encoders.** Fraction of held-out cells whose prediction leaves its old target. Deleting the captured combination relocates every cell of every seed on all three encoders, while deleting an irrelevant seen combination or merely redrawing the training seed does not. The re-seed floor is what makes the contrast readable, and it is low here: compare the 0.875 floor at language-model scale (Appendix I), where re-drawing the seed alone already moved the prediction and the test was therefore uninformative. *Posterior-mean read-out* (see the currency note below). 5 training seeds; the re-seed arm pools 25 re-draws.

frozen encoder	delete target	delete irrelevant	re-seed floor	target − re-seed
DINOv2-b	1.00	0.193	0.330	+0.670
SigLIP-b	1.00	0.073	0.119	+0.881
CLIP ViT-B/32	1.00	0.108	0.110	+0.890

What we claim here, and what we do not. Three qualifications, all of which narrow the result.

Currency. Section 4 is sampled throughout and we hold ourselves to that. The table above is the *posterior-mean* read-out, and we label it as such so the currencies stay separate. The reason is measurable: under a sampled read-out at this operating point these models stop fitting their own seen support, which is the regime in which the selection rule of Proposition 1 is not expected to apply. Sweeping the rate, sampled seen accuracy rises from 0.40 at $\beta{=}0.55$ to 0.93 at $\beta{=}0.05$, but held-out composition rises with it, and only one cell we measured, CLIP at $\beta{=}0.05$, both fits the seen support at ≥ 0.90 and still fails to compose. Predictability is the finding that does not depend on this: the nearest-seen predictor names the captured target at 0.66–0.68 on CLIP and SigLIP under a sampled read-out at every rate from $\beta{=}0.55$ down to $\beta{=}0.05$, against a random control below 0.01.

Concentration is not claimed here. Under sampling the modal mass sits within 0.02 of the product-of-marginals null at every rate measured. That is consistent with Section 4.1, where we argue the *amount* of concentration is a property of the factorised head, and we therefore rest nothing on it on this substrate.

Registration. This study was run without a pre-registration and is exploratory. The operating-point search reported in the currency note above was registered before it ran. We keep the two at separate standing.

I Does capture replicate at language-model scale? Mostly not

Our motivating examples are gist tokens and prompt compression, so the obvious question is whether capture, the whole three-part structure of Section 4, reproduces there. We ran it and report the answer, which is largely negative and which we think is more informative than a replication would have been.

Setup. Frozen Qwen2.5-1.5B, $K{=}4$ gist tokens behind a VIB rate bottleneck, with the decode pass masked so it never sees the instruction. The task is the operation(8)×format(6) factorial, 12 of 48 cells held out per seed, four paraphrase templates. Queries were re-filtered so that a generated string identifies at most one (op, fmt) pair, checked exhaustively over the 48 queries × 48 cells, so

the read-out is *probe-free*: we parse what the model actually generated. Three arms at matched achieved rate: shared, routed, and an **uncompressed control** that puts the instruction in context with no bottleneck. That third arm is the one that decides whether anything here is about compression, and it is the arm the image protocol does not have.

The budget does not bind. The frozen decoder only functions above $\sim$80 achieved nats; below that seen generation collapses with everything else. The operating point is 117.7 nats against the 3.871 nats of (op, fmt) content the instruction actually carries, a factor of 30. The constraint is active (the multiplier is positive and the KL tracks its target) but nowhere near information-theoretically tight. Nothing below can be attributed to a tight bottleneck.

P1, concentration: partial, and present without compression. Held-out cells are answered with a *seen* combination 0.615 of the time, and sharply: modal mass 0.720 $[0.665, 0.771]$, predicted-pair entropy 0.69 nats against $\log |S| = 3.584$, 0.957 of error mass on seen combinations against a 0.766 uniform null, and targets are cell-specific (0.72 distinct per cell; the largest single attractor takes 0.28). But the *uncompressed* control errs just as concentratedly (modal mass 0.798, entropy 0.43). Compression changes how *often* the failure takes this form (conditional on an error, the answer is a seen combination 0.81 of the time compressed against 0.45 uncompressed), while the concentration itself is present either way.

P1b, code geometry: failed, with the sign reversed. This is the sharpest disagreement with the image result. On images the held-out code sits three times *closer* to the nearest seen combination than to the additive point (ratio 0.33–0.36, with 0.95–1.00 of cells below 1). Here the ratio is 2.84 $[2.68, 2.99]$, the code sits nearly three times *farther* from the nearest seen combination than from the compositional point, and only 0.01 of cells fall below 1. A rate-aware read-out trained on seen cells recovers the correct held-out pair from the code 0.731 of the time. **The gist code is compositional; the failure happens downstream, in how the frozen decoder reads a compressed soft prompt.**

P2, predictability: partial, and fully explained by factor sharing. The frozen-Qwen-embedding-nearest seen instruction predicts the target at 0.214 against a 0.028 chance rate and a 0.028 random control. But the target shares exactly one factor with the held-out cell 0.95 of the time (operation 0.593 against a 0.103 chance rate), and *picking uniformly at random among seen cells that share the operation already scores* 0.165, while the embedding-nearest cell *within that same set* scores 0.154, no better. Token overlap and edit distance add nothing (0.158); degree is at 0.008. On images the perceptual-nearest predictor reaches 0.68–0.85 (sampled; 0.68–0.80 at the posterior mean) while "shares a factor value" is at chance, so the two settings are qualitatively different: there, similarity carries the signal; here, only factor structure does.

P3, causality: failed. Deleting the target from the support and retraining moves the prediction 0.958 $[0.875, 1.000]$, against 0.739 $[0.609, 0.870]$ for a matched control deletion. That $+0.22$ gap looks like the image result until one adds a third condition, which we did: **delete nothing and merely re-draw the training seed**. That still moves the prediction 0.875 $[0.750, 1.000]$. The graded version agrees, the probability of the old target falls 0.709 under target deletion, 0.557 under control deletion and 0.617 under no deletion at all, and with the target left in the support the retrained model returns to it only 0.13–0.26 of the time. The capture target here is not determined by the support; it is largely an accident of the training run. What is partly stable is factor-level structure: the new target keeps the old target's operation about half the time.

Why the image causal test is not subject to the same objection. The image control deletes an irrelevant seen combination and *retrains from scratch* at the same seed, so it already prices retraining variance, and it moves only 0.275 of predictions against 1.00 for target deletion. The LM instability is of a different order, and it is what makes the third condition necessary there.

What we take from this. Capture, as a mechanism with a predictable and support-determined target, is a property of the compressed codes we train, and we do not claim it for compressed interfaces in general. At language-model scale the code stays compositional and the analogous failure lives in the frozen decoder, is organised by factor structure with similarity playing little role, and is not reproducible across retrains. We report it as a boundary on the paper's motivating story. Two caveats

on our own numbers: the routed arm failed to fit the seen cells in three of four seeds (seen exact-match 0.12–0.18), so we make no arm comparison here, and its achieved rate ran 11% above shared (130.8 against 117.7 nats); and the uncompressed control's P2 row rests on 14 captured cells. Full records ship with the code release.

J ADDITIVE ANCHORING

The main text states the outcome; here is the protocol and the diagnostic behind it. On the plain code a learned operator over per-value prototypes reaches 0.91 and the closed-form GLS read-out matches it, while encoding a real held-out image reaches 0.00, so the failure is in where the encoder *places* held-out inputs. That invites the most direct attack we know of on our own mechanism: constrain the encoder's map itself with $\lambda\|z(x) - \sum_f p_f[y_f(x)]\|^2$, forcing the code to be additive in the factor prototypes. Additivity extrapolates off the support by construction, so the map should then place unseen combinations correctly. Selecting λ on a seen-only validation split at matched achieved rate, it does not work, and the reason is instructive: the penalty achieves exactly what it is written to achieve on the support (the on-support additivity gap falls from 0.19 to 0.07 on dSprites) while off the support the gap barely moves, from 2.40 to 1.42, still large enough that a held-out code sits farther from $\sum_f p_f$ than from the global mean. On the synthetic task the arm rescues a plain token (+0.26, 95% CI $[0.11, 0.41]$) but only ties per-block free bits and loses to routing; on both image benchmarks the leak-free protocol selected $\lambda = 0$, declining to anchor at all. A loss defined on the support does not reach the encoder's off-support behaviour.

Selection protocol. λ was selected on a validation split of *seen* combinations only, never on held-out cells, at matched achieved rate against every other arm. On both image benchmarks that leak-free protocol chose $\lambda = 0$: given only seen-combination evidence, the penalty does not look worth applying, which is itself the point. We report this arm because it is the most direct attack we know of on the mechanism we identify, and it fails for a reason the mechanism predicts: a loss defined at labelled points has no purchase on where unlabelled inputs are placed (Appendix E).

K FROZEN ENCODERS

This appendix supports the six-encoder result of Section 6, where tightening the budget costs a plain bottleneck three times what it costs a specialised one.

Table 20: **The tradeoff on frozen modern embeddings, at matched achieved rate.** 3DShapes object-hue × scale, six frozen encoders, 10 seeds, *posterior-mean* read-out (both arms; the contrast is within one currency). Both arms are swept over β and each is read at *common achieved rates* by interpolation along its own frontier, so nothing here depends on the two arms landing on the same rate. Tightening the budget from 12 to 4 nats costs the plain bottleneck 0.24 held-out composition, in *all six* encoders, while the routed one loses 0.08. The table reports interpolated means and the final column is computed on the unrounded frontier, so it can differ by 0.01 from the difference of the rounded cells. Per-seed uncertainty is available only at the measured operating points, so monotonicity should not be read as zero uncertainty.

	achieved rate (nats): plain / routed					plain
frozen encoder	12	10	8	6	4	loses
DINOv2-small	0.49/0.76	0.47/0.77	0.45/0.78	0.36/0.79	0.28/**0.66**	−0.22
DINOv2-base	0.62/0.84	0.59/0.85	0.55/0.86	0.46/0.87	0.34/**0.71**	−0.28
DINOv2-large	0.67/0.86	0.65/0.87	0.62/0.87	0.52/0.88	0.39/**0.70**	−0.28
CLIP ViT-B/32	0.28/0.45	0.26/0.47	0.25/0.49	0.18/0.55	0.12/**0.50**	−0.17
CLIP ViT-L/14	0.36/0.51	0.34/0.53	0.32/0.54	0.24/0.57	0.14/**0.48**	−0.22
SigLIP	0.46/0.63	0.43/0.64	0.40/0.65	0.31/0.66	0.21/**0.54**	−0.25
mean	0.48/0.68	0.46/0.69	0.43/0.70	0.35/0.72	0.24/**0.60**	**−0.24**

Routing makes factors recombinable. Splicing one factor's block from one example and the other's from a second, the routed heads return the swapped pair with no drift (recombine 1.00, drift 0.00), while a plain token cannot (0.33 over six random partitions).

L FROZEN LANGUAGE MODELS

The frozen-LM result of Section 6 uses Qwen2.5-1.5B. Two checks show it is neither model- nor scale-specific: we rerun the identical protocol (same templates, grid, held-out split, rate sweep, 3 seeds) with only the frozen backbone swapped. **Second family:** on SmolLM2-1.7B (a Llama-architecture model) the routed>shared gap reproduces and grows with compression, peaking at $+0.70$ (vs $+0.80$ on Qwen). **Scale:** on the base Qwen2.5-7B (7.6B parameters) the gap holds and if anything strengthens, peaking at $+0.87$, and the shared code collapses at the same ~ 10-nat achieved-KL budget as at 1.5B (Table 21). This empirical scale invariance is consistent with a rate-controlled effect, but is not predicted by either rate result, which contains no backbone-size variable. The gaps above are at matched β. Read post hoc at matched achieved rate, by per-seed interpolation along the routed arm's frontier at each shared point's rate (Figure 5), routing composes more at all 11 points where both arms reach the same rate, by $+0.05$ to $+0.13$ above 15 nats, $+0.46$ to $+0.66$ at 11–13 nats and $+0.57$ to $+0.69$ at 6–10 nats. The shared code's four lowest-rate points lie below any rate the routed arm reached. The routing advantage is a compression-regime effect: at high rate a large model composes under either scheme, and the gap opens only once the representation is squeezed toward the routing rate.

Table 21: Frozen Qwen2.5-7B (7.6B), held-out composition vs. compression (3 seeds). Shared compression collapses at the $\sim$10-nat budget while per-factor routing holds. The gap peaks at $+0.87$, matching the 1.5B pattern.

rate (nats)	4973	19	10.3	3.5	0
shared	0.90	0.85	0.05	0.00	0.00
routed	1.00	0.97	0.92	0.68	0.00
gap	$+0.10$	$+0.12$	$+\mathbf{0.87}$	$+0.68$	0.00

M GENERATIVE READ-OUT THROUGH A FROZEN DECODER

The generative test renders held-out combinations by swapping per-factor blocks through a frozen decoder. On canonicalised 3DShapes a conditional diffusion decoder (predicting the clean latent directly; naive ϵ-prediction collapses) renders held-out combinations at 0.454 ± 0.104 from a routed code against 0.087 ± 0.065 from a plain one, but the plain arm's seen-combination generation is only 0.329, so block-swap is not a valid operator for it and we read this as diagnostic only. Encoding CelebA into the frozen Stable-Diffusion KL autoencoder (Rombach et al., 2022), a routed code over its 128×128 latents composes held-out attribute combinations (smiling, male, eyeglasses) at 0.268 ± 0.058 against 0.003 ± 0.003 for a matched plain code, with the same caveat (0.778 vs. 0.277 seen-combination generation).

Fair-operator control, and how a pre-registered follow-up weakened our claim. Because block-swap is unfair to a plain code, we fit a composition operator per arm on seen cells only and evaluate on held-out cells (two combinations held out, chance 0.125). A single-seed smoke run at 2,500 steps put the plain arm's learned operator at 0.000 held-out; we registered a three-seed replication at the full 16,000 steps with two pre-declared outcomes, no rescue (≤ 0.10) or full rescue (≥ 0.214). The outcome fell *between* them and we report it as registered. At full training the plain code is no longer at zero: its best learned operator reaches 0.147 ± 0.207 (linear) and 0.099 ± 0.140 (MLP), driven by a single seed each (per-seed $0.000/0.440/0.000$ and $0.000/0.000/0.296$). The routed arm stays ahead on every operator (block-swap 0.264 ± 0.065; learned MLP 0.499 ± 0.407; matched reconstruction 0.957 vs. 0.952). We therefore no longer claim that a plain code *cannot* support held-out generative composition on real photographs under an arm-appropriate operator: it partially can. Specialisation makes composition available fit-free and roughly doubles what the best fitted operator extracts. The three-seed variances are large and we do not read the operator-level ordering as established.

Generative capture. The same placement failure appears when the decoder must *render* a held-out combination. On canonicalised dSprites and 3DShapes a plain VAE's wrong generations land in the seen support above a uniform-wrong null and concentrate on the code-nearest seen cell (pre-registered tests G1–G3 all passed; G1 in 2 of 3 seeds on 3DShapes, every other gate in 3 of 3). Mean held-out render accuracy is 0.058 (dSprites) and 0.062 (3DShapes) for the plain code against 0.602 and 0.666

for the routed code. Capture is therefore not an artefact of a classification head: the decoder itself snaps to a seen combination. One caution on the metric: these are feed-forward numbers scored by a frozen attribute classifier, and *optimising* an assembled code at test time against that scorer drives its reported accuracy to ≈ 1.0, above the real-image ceiling, while an independent DINOv2 judge does not follow (≈ 0.20). Compositional-generation accuracy should be read off feed-forward samples with a judge not used in any optimisation.

N SUPPORT GEOMETRY: THEORY AND THE PRE-REGISTERED TEST

This appendix proves Proposition 1 of Section 4.2, maps where it holds, reports the registered before-training test of support geometry, and develops an explicit surrogate model.

N.1 WHICH SEEN CELL, AND WHEN: PROOF OF PROPOSITION 1

Setting. Fix a code z. The seen cells $s \in S$ have cluster means m_s and a common isotropic covariance $\sigma^2 I$; write $d_s = \|z - m_s\|$ and order the cells so that $d_{s_1} \le d_{s_2} \le \dots$, with $\Delta = d_{s_2}^2 - d_{s_1}^2 \ge 0$. The *support-conditioned* read-out assigns z a responsibility over seen cells, $r_s(z) = \exp(-d_s^2/2\sigma^2) / \sum_{s' \in S} \exp(-d_{s'}^2/2\sigma^2)$, and reads factor f by marginalising, $p(y_f{=}i \mid z) = \sum_{s:\, s_f = i} r_s(z)$. This is the Bayes posterior of y_f under an equal-prior Gaussian mixture on the seen cells; it is the population optimum of the per-factor cross-entropy over all read-outs, and a linear-softmax head is its best approximant in that class. Unequal priors π_s replace d_s^2 by $d_s^2 - 2\sigma^2 \ln \pi_s$ throughout.

Factorisation. For any per-factor heads, $p(y \mid z) = \prod_f p(y_f \mid z)$ holds for every z by construction, so the modal joint mass at z is the product of the marginal modes, and the heads can express no coupling between factors beyond what both read from z. The empirical statement in Section 4.1, 0.818 against 0.830 (0.874 against 0.886 at the posterior mean), is the aggregate over the inputs of a held cell and says that this per-code factorisation survives averaging, i.e. the heads are near-deterministic on those inputs.

Proof of (i). $r_{s_1} = \big(1 + \sum_{s \ne s_1} e^{-(d_s^2 - d_{s_1}^2)/2\sigma^2}\big)^{-1} \ge \big(1 + (|S|-1)e^{-\Delta/2\sigma^2}\big)^{-1}$, which exceeds $\frac{1}{2}$ iff $(|S|-1)e^{-\Delta/2\sigma^2} < 1$, i.e. $\Delta > 2\sigma^2 \ln(|S|-1)$. Now fix f and let $i^\star = (s_1)_f$. Then $p(y_f{=}i^\star \mid z) \ge r_{s_1} > \frac{1}{2}$, while any $i \ne i^\star$ collects only cells other than s_1, so $p(y_f{=}i \mid z) \le 1 - r_{s_1} < \frac{1}{2}$. Hence $\arg\max_i p(y_f{=}i \mid z) = (s_1)_f$ for every f, and the modal joint prediction is s_1. Nothing in the argument asks whether s_1 is the true label of z. □

Proof of (ii). Deleting s_1 and refitting the same read-out on $S' = S \setminus \{s_1\}$ leaves the cluster means of the remaining cells unchanged by assumption; the responsibilities renormalise over S', whose nearest element to z is s_2. Part (i) applied to S' with gap $d_{s_3}^2 - d_{s_2}^2$ gives modal prediction s_2; induction gives s_{k+1} after deleting $\{s_1, \dots, s_k\}$. □

Proof of (iii). Let $z = (z_1, \dots, z_F)$ and suppose $m_s = (m_{s_1}^{(1)}, \dots, m_{s_F}^{(F)})$, i.e. block f of a seen-cell mean depends on the cell's value of factor f alone (the direct-sum structure that routing encourages). A head that reads only z_f sees the value clusters $\{m_i^{(f)}\}_{i \in [V_f]}$ and its Bayes read-out sums the responsibilities of every seen cell carrying that value. All of them share the same block-f mean, so the sum is a count times one Gaussian, $p(y_f{=}i \mid z) = p(y_f{=}i \mid z_f) \propto \deg_S(i) \exp(-\|z_f - m_i^{(f)}\|^2/2\sigma^2)$ with $\deg_S(i) = |\{s \in S : s_f = i\}|$. An equal prior over *cells* is not an equal prior over *values*, and under a partial support the counts need not match. Its argmax is therefore the minimiser of $\|z_f - m_i^{(f)}\|^2 - 2\sigma^2 \ln \deg_S(i)$, which is the nearest value prototype in block f exactly when $\deg_S$ is constant in i, or when the block margin exceeds $2\sigma^2 \ln(\deg_S(j)/\deg_S(i))$ for every rival j. The modal joint prediction is the tuple of per-block nearest values, a cell of $\prod_f [V_f]$ that is not constrained to lie in S. Since $p(y_f \mid z)$ does not depend on $z_{f'}$, $\partial \log p(y_f \mid z)/\partial z_{f'} = 0$ identically. Under the shared read-out of (i), $\partial \log p(y_f{=}i \mid z)/\partial z_{f'} = \sum_{s:s_f=i} r_s\,(m_s - z)_{f'}/\sigma^2 / p(y_f{=}i \mid$

$z) - \sum_s r_s(m_s - z)_{f'}/\sigma^2$, which vanishes for all z only if the cells sharing value i of f have the same block-f' mean as the full support, i.e. only in degenerate supports. □

What is assumed, and what the experiments test. Three assumptions do work, and a fourth is specific to (iii). (a) The read-out is fit to the seen clusters and nothing else, which is what training on S provides. (b) Within-cell spread is common and isotropic. That is an idealisation. On the dSprites capture rows a held cell's mean code sits a median 2.3 within-cell standard deviations from the nearest seen mean, so the held code sits at the edge of the nearest cluster, and (i) then turns on the gap Δ between the two nearest cells rather than on any one cluster's tail. (b′) For the nearest-*value* reading of (iii), either the support degrees are equal or the block margin clears $2\sigma^2 \ln(\deg_S(j)/\deg_S(i))$. Our splits are not degree-balanced, the dSprites scale values occurring in 1 to 3 seen cells, so this condition has real content, and the cross-factor derivative half of (iii) does not depend on it. (c) *Placement*: the proposition takes z as given. Which s_1 a held-out input has is decided by the encoder, which the surrogate does not model, and the pixel predictor of Section 4.2 is the empirical statement that placement approximately preserves input proximity. The deletion experiments therefore test (ii) *jointly* with placement stability: they retrain the encoder as well as the read-out, so a cascade that follows the distance ordering says that the encoder's placement of a held-out input is itself stable under support edits, a stronger statement than (ii) alone. Part (iii) is why routing reduces capture, and its cross-factor derivative is the population version of the crossed probe of Appendix G, which is the dominant predictor of composition across arms ($r = -0.73$). One negative belongs here: the margin Δ of (i) is defined in the read-out metric, and its raw-pixel proxy does not predict capture *strength* (Spearman with modal mass 0.18 $[-0.20, 0.57]$ on dSprites, 0.02 on Colored-MNIST). Pixel distance carries the address but not the margin.

The proposition on this paper's own codes. Pre-registered. The proposition is verified numerically on random cluster geometries, which tests the algebra and not this paper's encoders. Here its read-out is applied to trained dSprites codes instead, β=0.01, 10 seeds, with no training of any kind. Prototypes are seen-cell mean posterior codes and σ^2 is the pooled within-seen-cell variance, both measured as above. The dominance condition of (i) holds on 0.92–0.97 of held-out cells. Deleting the modal seen cell and recomputing moves the prediction on 0.974 $[0.946, 1.000]$ of cells, the new prediction is the nearest remaining seen cell on 0.887 $[0.804, 0.958]$, and iterating holds to depth three at 0.946 $[0.865, 1.000]$. Part (ii) therefore holds on the codes this paper trains. Two limits are worth stating. Scored against the *pixel* ordering rather than the code ordering, this frozen-encoder cascade reaches only 0.235 $[0.174, 0.292]$ where the retrained cascade of Section 4.3 reaches 0.704, although its first address matches the 0.761 of Section 4.2 at 0.700. The address of the first capture is already in the frozen code and the ordering after a deletion is not, which is what retraining supplies, and a post-hoc sweep of twenty read-out variants (a 16-fold σ^2 range, pooled against per-dimension variance, two aggregations) moves that 0.235 only to 0.258, so it is a property of the code and not of the read-out. The support-degree correction of (iii) changes no argmax in 80 held-out cell-factor pairs, the distance gap exceeding $2\sigma^2 \ln(\deg_S(j)/\deg_S(i))$ by a median factor of 135 and a worst case of 2, which is why (b′) stays stated as a condition.

The formula tested as a number. Pre-registered. dSprites shape×scale, plain arm, five rates × five seeds, four held-out cells each (100 cells). For every held-out cell the predicted modal mass is the mean over the cell's images of the responsibility vector $r_s(z)$ over seen cells, then the maximum; prototypes m_s are seen-cell mean posterior codes and σ^2 is the pooled within-seen-cell code variance averaged over latent dimensions, both measured, nothing fitted. The observed modal mass is the trained heads' joint (product of the two softmaxes), averaged over the same images, restricted to seen cells. Gates were scored on the 80 cells whose model retains in-support accuracy ≥ 0.8.

Which quantity is being validated. A factorised read-out assigns the joint $\prod_f \sum_{s:s_f=(s_1)_f} r_s$, which differs from r_{s_1} by exactly $ab - rc$ where a, b are the responsibility leaked to seen cells sharing one of s_1's values and c to cells sharing neither. r_{s_1} is therefore a lower bound on each head's marginal rather than the joint itself. Measured on the same 80 cells, the factorised joint tracks the trained heads at $r = 0.817$ $[0.711, 0.892]$ against 0.830 for r_{s_1}, a paired difference of -0.014 $[-0.061, 0.029]$, so the distinction does not separate them here. Both overshoot the heads' confidence, r_{s_1} by 0.100 and the factorised joint by 0.071, which is the sense in which this prices selection and not strength.

Table 22: Proposition 1(i) as a quantitative prediction. Per rate: in-support accuracy, measured noise σ^2, median dominance ratio $\Delta/2\sigma^2 \ln(|S|-1)$, and predicted against observed modal mass (20 cells each). †: in-support accuracy collapsed; excluded from the gates by pre-registration and shown to mark where the formula stops applying.

β	seen acc.	σ^2	median ratio	predicted	observed	pred.−obs.
0.001	0.968	0.3070	31	0.939	0.889	+0.050
0.01	0.944	0.0656	35	0.870	0.802	+0.068
0.1	0.968	0.0149	42	0.874	0.745	+0.128
0.3	0.938	0.0045	88	0.843	0.689	+0.154
$1^{\dagger}$	0.242	0.0003	16	0.742	0.105	+0.637

Over the 80 gated cells: Pearson $r = 0.830$ (seed-cluster bootstrap $[0.762, 0.884]$), Spearman $\rho = 0.883$; mean absolute error 0.110 against 0.176 for the no-information constant (gain 0.066, CI $[0.043, 0.082]$); calibration slope 1.12, intercept -0.21; predicted captured cell equals observed in 93.8%. Controls: rank correlation with observed modal mass is 0.708 for Δ alone and 0.269 for d_{s_1} alone. Isotropy diagnostic: the ratio of largest to smallest per-dimension within-cell variance has median 37 and maximum 373. Gate G3 (phase boundary) is *not evaluable*: all 80 cells have ratio > 1 (range 3.2–231), and even per image only three cells have more than 10% of images below the threshold.

The boundary is unreachable. It is fair to ask whether some other support would put cells on the boundary, and the answer is no for any support one could train on. Two quantities set the ratio $\Delta/2\sigma^2 \ln(|S|-1)$. The first is nearly inert: σ^2 falls four orders of magnitude across our rate grid (0.307 at β=0.001 to 0.0003 at β=1), but Δ scales with the same code, so the median ratio stays between 31 and 88 at *every* rate where the support is still decoded. Rate is not a lever. The only remaining lever is $\ln(|S|-1)$, and it is logarithmic. Enlarging the support by adding factors is the fastest way to grow it: a model-free calculation over factored supports with F up to 6 gives $\ln(|S|-1)$ rising from 3.3 to 8.1, which makes the boundary $2.1\times$ easier to reach ($3.4\times$ for the closest-spaced quartile of cells) while the *relative* margin $(d_2 - d_1)/d_1$ stays flat at 0.13–0.21. Applying that factor to every measured cell moves none of the 80 below threshold on median geometry and 1.2% on the most favourable. Closing a gap of 31–88 needs roughly $30\times$, hence $\ln(|S|-1) \approx 100$ and a support of order e^{100} cells. The dominance condition is therefore satisfied throughout the regime in which these models work, and violated only where the seen support is no longer decoded, which part (i) already excludes. Proposition 1(i) should be read as a description of the operating regime, whose boundary no trainable model sits near. It is a sufficient-condition model of *selection*, and it sets no quantitative law of capture *strength*. The calculation assumes a generic factored embedding in place of a trained encoder, which is why we quote it as a two-order-of-magnitude verdict and not a tight number.

A registered prediction that failed. The addendum registration predicted that the dominance ratio would *negatively* predict composition success. As registered, with joint held-out accuracy as the outcome, the test is degenerate: that accuracy is 0.000 on every cell. A post-hoc re-test with mean marginal head accuracy, the quantity behind the $\rho = +0.44$ distance–success correlation that Section 4.2 mentions and the one part (i) speaks to, is disclosed as post-hoc and also fails, with the *wrong sign*: $\rho = +0.305$ $[+0.061, +0.544]$ against $+0.270$ for raw distance, which the ratio was meant to improve on. This is why the main text no longer attributes the distance–success correlation to cells failing the dominance condition: none does.

The validity map. Figure 11 places every measured cell against the proposition's assumptions. Where the support is decoded (seen accuracy 0.94–0.97) the formula predicts the observed modal mass at $r = 0.83$ with nothing fitted, *despite* the isotropy assumption being violated in every cell; where the support is no longer decoded (β=1, seen 0.24) it fails outright, as it should, since it prices a read-out of seen-cell clusters that no longer exist. The dominance condition itself is satisfied in every cell by a factor of 3 to 231, so the phase boundary the condition names is never crossed here. That is not an accident of this benchmark, and the next paragraph shows it cannot be.

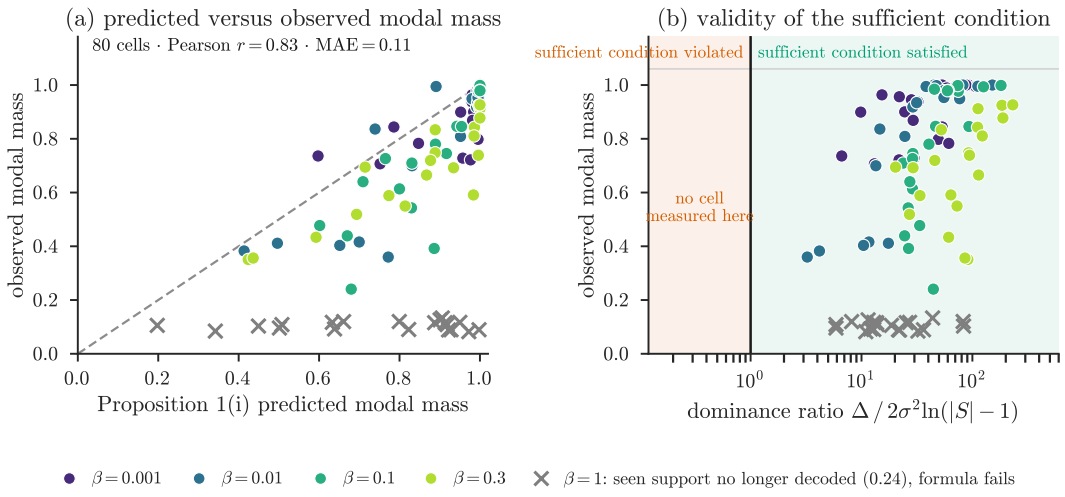

Figure 11: **Where Proposition 1(i) predicts, and where it fails.** dSprites shape×scale, plain arm, 100 held-out cells (five rates × five seeds). **(a)** Observed modal mass against the value Proposition 1(i) predicts with nothing fitted; the dashed diagonal is exact agreement. Over the 80 cells whose seen support the encoder still decodes, $r = 0.83$ and the mean absolute error is 0.11. **(b)** The same cells against the dominance ratio $\Delta/2\sigma^2 \ln(|S| - 1)$ on a log scale, with the vertical line at the sufficient-condition threshold. Three limits are visible. The condition is satisfied by every cell we measured, the smallest ratio anywhere being 3.2, so the shaded violating region is empty and that boundary is never probed. The isotropy the proof assumes is violated throughout (per-dimension variance ratio, median 37) and the prediction survives it. And the grey crosses are the β=1 cells, where seen-support accuracy has fallen to 0.24: there the formula fails outright (error 0.64) *while still satisfying the condition*, which is what it means for the condition to be sufficient for selection rather than for accuracy.

N.2 The pre-registered test

Capture is by the nearest seen neighbour, so the geometry of the seen support should predict which held-out cells are hard *before* any model is trained. We tested that prospectively on three real attribute×object benchmarks. For each, a prediction was computed from the split alone, with no accuracy of any kind, and committed read-only. The outcome record carries the SHA-256 of the prediction it tested and the UTC time it was written, so the ordering can be audited.

The primary statistic is the Spearman correlation between per-cell held-out accuracy and the cell's statistical *leverage* in the seen support (Definition 2), with cluster-bootstrap intervals over objects. The pre-registered bar was $\rho \leq -0.2$.

Table 23: The three pre-registered outcomes. "Bar" was fixed in advance at $\rho \leq -0.2$. The predictor met it on one benchmark, inverted on a second, and was correctly signed but too weak on the third. We report all three and claim only a weak geometric predictor, too weak for a per-cell forecast.

benchmark	cells	ρ(acc, leverage)	95% CI	pre-registered verdict
UT-Zappos50K	60	$\mathbf{-0.239}$	$[-0.423, -0.017]$	pass
Fashion-Product-Images	40	$+0.432$	$[+0.051, +0.651]$	wrong sign
MIT-States	1550	-0.103	$[-0.150, -0.050]$	null (right sign, under bar)

What replicates is which statistic carries the signal. On MIT-States, the largest of the three by a factor of 25, the effect is small but its *structure* is the one the theory below predicts. Leverage partialled on a per-value degree count remains -0.116 $[-0.191, -0.058]$, while the count partialled on leverage *flips positive* ($+0.103$ $[+0.051, +0.177]$). The Nash–Williams excess term, which is the part of effective resistance that is not explained by the two endpoint degrees, is the strongest single predictor at -0.164 $[-0.235, -0.085]$. So the geometric term survives conditioning on counting, and counting does not survive conditioning on geometry. The within-split permutation test is at its floor, $p < 5 \times 10^{-4}$ with 2000 resamples, and the sign is negative in all 5 splits.

How we read this. One pass of three is a weak prospective record and we say so in the main text. Support geometry is the right *kind* of statistic, and it survives controls that the obvious count-based alternative does not, while its effect size on real photographs is too small to forecast individual cells. The theory below explains why leverage is the quantity that should appear.

What was registered, and when. Each benchmark has a prediction written before its outcome was measured, with the analysis script's hash recorded in the registration (Zappos at 15:45 on 2026-09-01, Fashion-Product-Images at 16:02 and MIT-States at 16:10 the same day). All three ship with the code.

N.3 WHY LEVERAGE: AN EXPLICIT SURROGATE

The preceding subsection established empirically that support geometry orders cell-specific operational collapse points. This appendix analyses that ordering within an explicit surrogate model of a finite-rate factored code. We prove that (i) a held-out cell is identifiable from the seen support exactly when its factor values are connected in that support, and (ii) for identifiable cells the minimax squared-error risk of *any* composition read-out is σ^2 times the cell's statistical leverage, which for two factors is the effective resistance between the cell's two values. Proposition 2 is logically prior: it gives an information floor for reliable full-product recovery. This appendix bounds the *excess* a particular cell requires within its explicit surrogate.

The surrogate, and what it is a model of. A factor-preserving additive code assigns prototypes and represents a combination as their sum. It idealises the direct-sum structure encouraged by routing.

Definition 1 (Additive-plus-noise surrogate $\mathcal{M}(\sigma)$)**.** There exist prototypes $\{\mu_v\}, \{\nu_u\}$ such that for every seen combination $(v', u') \in S$ the observed code is $b_{v'u'} = \mu_{v'} + \nu_{u'} + \varepsilon_{v'u'}$ with $\varepsilon \sim \mathcal{N}(0, \sigma^2 I_D)$ i.i.d. across cells of S. A *composition read-out* maps the seen observations to an estimate of $\theta = \mu_v + \nu_u$ for (v, u) held out.

The rate enters through the noise. A code trained at finite rate cannot realise additive structure exactly, and we summarise the per-cell residual as isotropic noise of scale σ decreasing in the excess rate. The model assumes that this noise is independent across seen cells and has a common isotropic σ. Neither holds exactly for a trained encoder, and the calibration below shows where this is visible. Setting $p_a = \mu_a$ on one side and $p_a = -\nu_a$ on the other, each seen observation is a noisy *potential difference* across an edge of the support graph G_S, and composition is the classical problem of estimating a potential difference in a resistor network. Effective-resistance control of estimation error is established in the pairwise-comparison literature (Hendrickx et al., 2019); our use is to map a compositional support design to that classical problem and test the resulting cellwise predictor. Let L be the graph Laplacian and $R_{\text{eff}}(a, b) = (e_a - e_b)^\top L^+ (e_a - e_b)$ when a, b share a component of G_S, with $R_{\text{eff}}(a, b) = +\infty$ otherwise.

Lemma 1 (Gauge; identifiability)**.** *Two additive solutions agree on S if and only if, on each connected component K of G_S, there is a constant c_K with $\mu'_v = \mu_v + c_K$ and $\nu'_u = \nu_u - c_K$. Consequently a held-out cell (v, u) is identifiable from S if v, u lie in the same component, and if they lie in different components the target shifts by the arbitrary $c_K - c_{K'}$ across equally valid solutions. Such a cell is non-identifiable and has unbounded worst-case squared-error risk unless a prior is added.*

Proof. Work per coordinate. If two solutions agree on S, their differences satisfy $\Delta\mu_{v'} + \Delta\nu_{u'} = 0$ on every edge; propagating along any path within a component forces $\Delta\mu = c_K$, $\Delta\nu = -c_K$, and consistency around cycles is automatic since every edge imposes the same relation. Conversely any such shift preserves the constraints. In potential form this is $\ker L = \operatorname{span}\{\mathbf{1}_K\}$. Then $\mu_v + \nu_u$ picks up $c_K - c_K = 0$ within a component and $c_K - c_{K'}$ across, which ranges over $\mathbb{R}^D$. For a cell across components, shifting the potentials by t on the component of v leaves every seen observation unchanged and moves the target by t, so every read-out has risk at least $\|t\|^2/4$ at one of the two solutions, and letting $\|t\| \to \infty$ makes the worst-case risk unbounded. □

Write B for the $|S| \times (V_1 + V_2)$ signed incidence matrix of G_S, whose row for a seen combination (v', u') is $e_{v'} - e_{u'}$, and $\mathbf{b}$ for the observed codes stacked in the same order, so that $L = B^\top B$.

Theorem 1 (Achievability). *In $\mathcal{M}(\sigma)$ let (v, u) be a held-out cell with v, u in a common component and $w = e_v - e_u$. The GLS read-out $\widehat{\theta} = w^\top L^+ B^\top \mathbf{b}$ is unbiased with $\mathrm{Cov}(\widehat{\theta}) = \sigma^2 R_{\mathrm{eff}}(v, u) I_D$, and it is the best linear unbiased read-out.*

Proof. With p the potentials, $\mathbb{E}\mathbf{b} = Bp$, so $\mathbb{E}\widehat{\theta} = w^\top L^+ Lp$. The matrix L^+L projects onto $(\ker L)^\perp$, and w lies there because v, u share a component, so $\mathbb{E}\widehat{\theta} = w^\top p = \mu_v + \nu_u = \theta$. Per coordinate the variance is $\sigma^2 w^\top L^+ B^\top B L^+ w = \sigma^2 w^\top L^+ w = \sigma^2 R_{\mathrm{eff}}$, and the coordinates are independent. The Gauss–Markov theorem for the estimable function $w^\top p$ gives the last claim. □

Theorem 2 (Exact minimax risk). *In $\mathcal{M}(\sigma)$, for any identifiable held-out cell, $\inf_{\widehat{\theta}} \sup_p \mathbb{E}[(\widehat{\theta} - \theta)^2] = \sigma^2 R_{\mathrm{eff}}(v, u)$ per coordinate, the infimum over all measurable read-outs, and the GLS read-out of Theorem 1 attains it.*

Proof. Achievability is Theorem 1. For the converse, let $\varphi = L^+ w$ be the unit-current potential and $\psi = \varphi / R_{\mathrm{eff}}$, so $\psi^\top w = 1$ and the Dirichlet energy is $\psi^\top L \psi = 1/R_{\mathrm{eff}}$: moving p to $p + t\psi$ shifts the target by exactly t at seen-support energy t^2/R_{eff}. Restrict to this one-dimensional subfamily and place a Gaussian prior $t \sim \mathcal{N}(0, \tau^2)$. The sufficient statistic for t is Gaussian with variance $\sigma^2 R_{\mathrm{eff}}$, so the Bayes risk of the posterior mean is $\sigma^2 R_{\mathrm{eff}}\, \tau^2/(\sigma^2 R_{\mathrm{eff}} + \tau^2)$. Minimax risk is at least the Bayes risk of any prior, and letting $\tau \to \infty$ gives $\inf \sup \geq \sigma^2 R_{\mathrm{eff}}$. With Theorem 1's matching upper bound the two coincide. □

What the surrogate does not model, tested directly. The surrogate models a read-out of a code already assumed additive; it says nothing about where a trained encoder *places* an input it never saw. A natural candidate for that gap is linearised kernel interpolation: if learning stays in the lazy regime, a held-out code is $\sum_{s \in S} k(x_h, x_s)\alpha_s$ for the network's own kernel k, and capture would be the statement that those weights are peaked, which would make concentration, predictability and re-capture consequences of one equation. We tested its one ex-ante commitment, that the network's own kernel places a held-out input better than raw perceptual distance, on the trained dSprites models. It fails: the post-training empirical NTK (full parameter set) picks the observed capture target in 0.350 $[0.150, 0.600]$ of cells against pixel distance's 0.800 on the same cells (paired -0.450 $[-0.750, -0.150]$), kernel peakedness does not track modal mass ($\rho = 0.237$ $[-0.342, 0.672]$), and the framework predicts the wrong sign for the support-distance correlation it was meant to explain. Cosine-normalising the kernel reaches 0.950, but its argmax coincides with the trained-code nearest cell in 0.85 of cells, so that number is the code predictor we already report. Capture is therefore not a lazy-regime phenomenon, and the placement question remains open: pre-registration and full variants in the code release.

The surrogate results hold only inside the additive class. A code that simply packs the seen support fits S at lower rate and fails on every held-out cell regardless of resistance. Proposition 2 compares coding costs, and Theorems 1 and 2 govern how well additive structure reaches a specific cell.

Remark on rate. Suppose the excess rate $\Delta R \geq 0$ beyond the floor is spent on prototype precision, so that under the Gaussian rate–distortion idealisation $\sigma^2 = \sigma_0^2 e^{-2\Delta R/D}$. By Theorem 2 the risk $\sigma^2 R_{\mathrm{eff}}$ then falls to a fixed level c at $\Delta R = \frac{D}{2} \ln(R_{\mathrm{eff}}/R_0)$ with $R_0 = c/\sigma_0^2$, and needs no excess rate when $R_{\mathrm{eff}} \leq R_0$. A cell with larger resistance therefore needs at least as much rate, and by Lemma 1 a cell across components is never recovered at any rate. Our sweeps measure achieved KL, so we test only this ordering and leave the slope $D/2$ untested.

Beyond two factors: composability leverage. Pairwise resistance is undefined for $F \geq 3$, but the estimation view generalises: the additive model is a regression on one-hot designs and resistance is a *leverage score*.

Definition 2 (Composability leverage). With A_S the design whose row for a seen combination is $a_s = \bigoplus_f e_{v_f}$ and $M = A_S^\top A_S$, set $\ell_S(h) = a_h^\top M^+ a_h$ if $a_h \in \mathrm{row}(A_S)$ and $+\infty$ otherwise.

Proposition 3 (Leverage generalises resistance). *In the F-factor surrogate, let a_h be the design row of a held-out cell. (i) The cell is identifiable iff $\ell_S(h) < \infty$. (ii) If it is, the GLS read-out $a_h^\top M^+ A_S^\top \mathbf{b}$ is unbiased with per-coordinate variance $\sigma^2 \ell_S(h)$, and the minimax squared-error risk over all read-outs is exactly $\sigma^2 \ell_S(h)$ per coordinate. (iii) At $F = 2$, $\ell_S(h) = R_{\mathrm{eff}}(v, u)$.*

Proof. (i) A linear function $a_h^\top P$ of the stacked prototypes P is determined by the seen data $A_S P$ exactly when $a_h \in \mathrm{row}(A_S)$. Otherwise there is $z \in \ker A_S$ with $a_h^\top z \neq 0$, and shifting P along z leaves the data unchanged while moving the target, so the worst-case risk is unbounded as in Lemma 1. (ii) Repeat the proofs of Theorems 1 and 2 with M in place of L and a_h in place of w. Here M^+M projects onto $\mathrm{row}(A_S)$, and $\psi = M^+ a_h / \ell_S(h)$ satisfies $\psi^\top a_h = 1$ and $\psi^\top M \psi = 1/\ell_S(h)$. (iii) With $T = I_{V_1} \oplus (-I_{V_2})$, row by row $A_S T = B$, so $L = TMT$. Since T is orthogonal and its own inverse, $M^+ = TL^+T$, and $Ta_h = e_v - e_u = w$ gives $\ell_S(h) = w^\top L^+ w = R_{\mathrm{eff}}(v, u)$. Also $a_h \in \mathrm{row}(A_S)$ exactly when $w \in \mathrm{row}(B) = (\ker L)^\perp$, that is when v, u share a component, so the two are infinite together. □

Leverage is computable from the split alone before training. Non-uniform supports need no separate treatment. When a seen cell has n_s observations, R_{eff} and ℓ_S are computed with edge weight n_s in L and row weight n_s in M.

The prediction that distinguishes it, tested. At $F = 2$ leverage *equals* resistance, so the two cannot be separated there, and in the dense supports we use both are close to the simple per-value count $1/\deg(v) + 1/\deg(u)$: on dSprites the three correlate at 0.94 and the partial correlation of resistance given degree is $+0.06$, i.e. the graph structure beyond degree does no work at this density. The theory's content is therefore at $F \geq 3$, where leverage and pairwise summaries differ. On the three-factor 3DShapes sweep (447 identifiable held-out cells) leverage attains pooled Spearman -0.53 against per-cell composability, compared with -0.40 for clique-expansion resistance and -0.52 for the per-value count. Because these cells come from only 12 shared-support runs, cell-level conditional intervals are not confirmatory; the mean within-run correlation is -0.291, and -0.24 once degree is partialled out. We therefore claim that leverage supplies a principled multi-factor diagnostic. This sweep does not establish an advantage over degree.