COLAB NOTEBOOK :
https://colab.research.google.com/drive/1yUAGtnTA8rvHFbVvQp0qpXew5qNWbCFr?usp=sharing

How a 10-Neuron Network Serves 100 Features

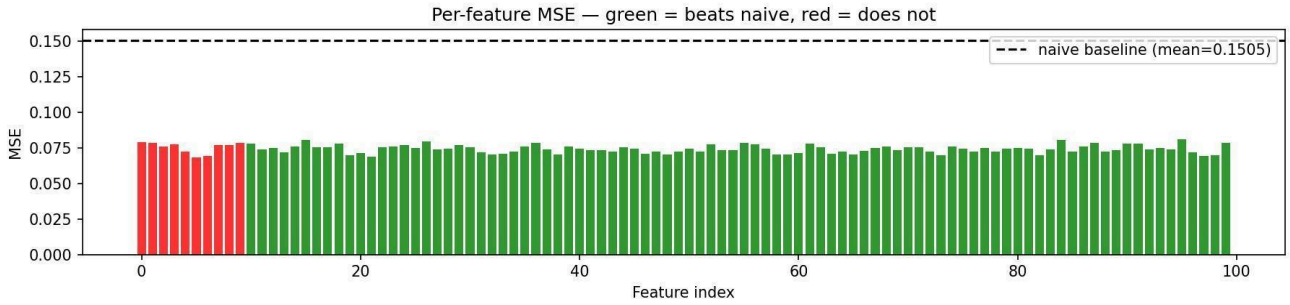
Model: One-hidden-layer MLP, no bias, no skip connection. 100 input features, 10 hidden neurons, 100 outputs. Trained with L4 loss to reconstruct ReLU(x) from sparse inputs (each feature independently active with probability p = 0.02).

The puzzle: With only 10 neurons for 100 features, the network has no way to dedicate one neuron per feature. What does it learn instead?

TASK (part 1)

Does it even work?

The naive baseline dedicates each of the 10 neurons to one feature, it gets those 10 nearly perfect and outputs zero for the other 90. The trained model was given the same architecture but optimized end-to-end with L4 loss.



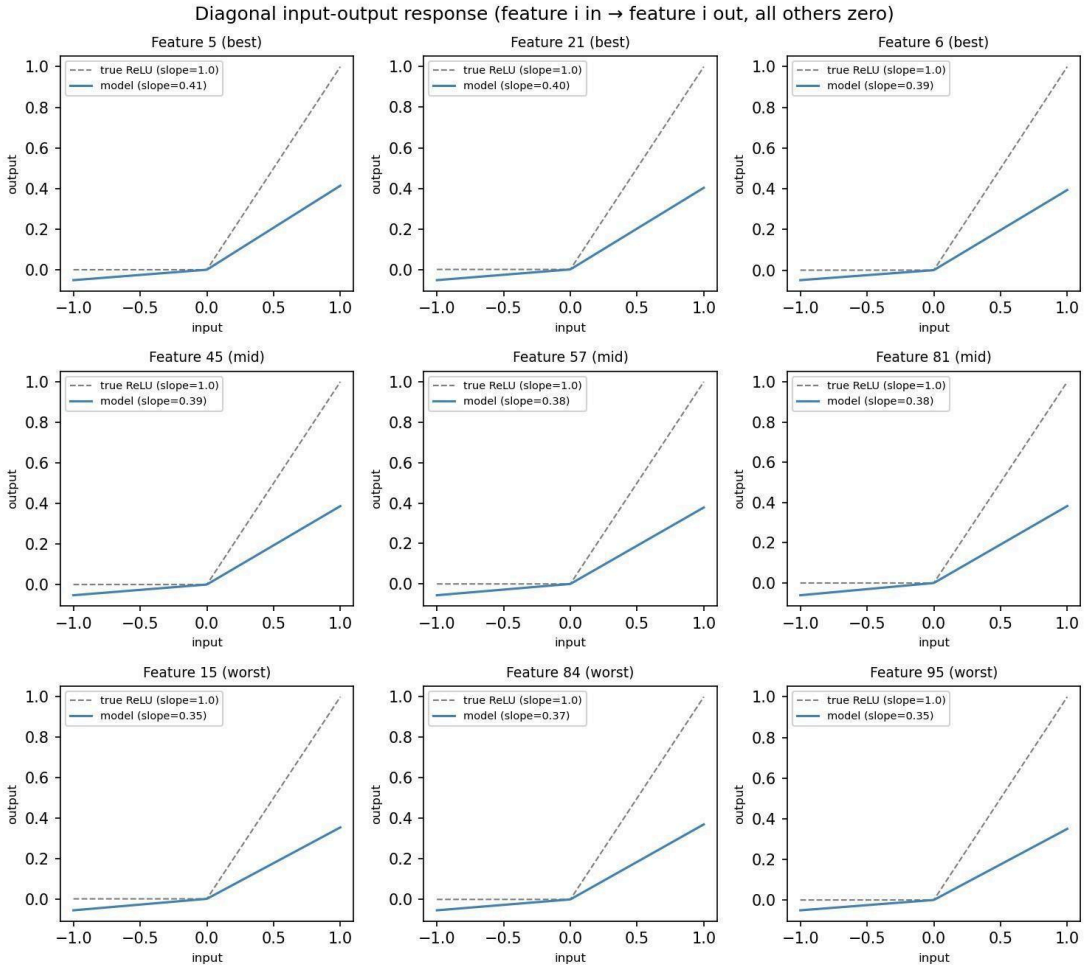
Per-feature MSE, green = beats naive baseline, red = does not. Naive baseline mean = 0.1505.

The trained model beats the naive baseline on 90 out of 100 features. **Mean MSE: 0.0742 (trained) vs 0.1505 (naive)**, roughly a 2× improvement. The 10 red bars on the left are features 0-9, exactly the ones the naive solution gets for free. The trained model gave those up in exchange for partially covering all 100 features. The L4 loss made that a worthwhile tradeoff: L4 penalizes large errors as x^4, so completely ignoring 90 features is far more costly than having moderate error on all of them.

What does the response look like?

The natural next question: what does “partially covering” mean? Is the model outputting a scaled ReLU, or something noisier?

COLAB NOTEBOOK :
https://colab.research.google.com/drive/1yUAGtnTA8rvHFbVvQp0qpXew5qNWbCFr?usp=sharing



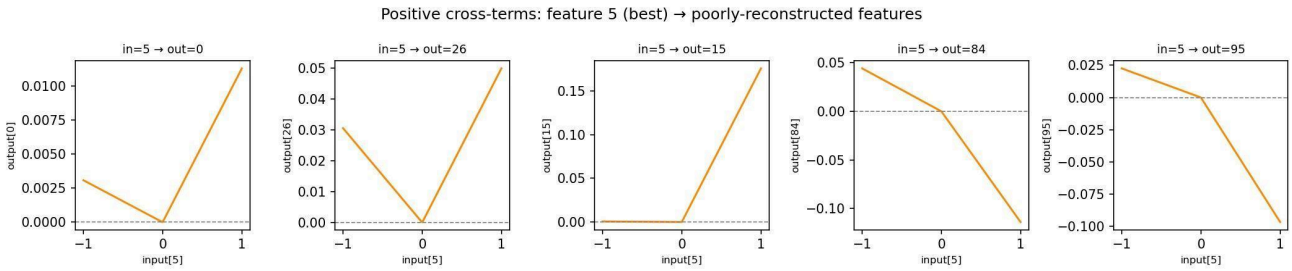
Diagonal input–output response (feature i in -> feature i out, all others zero). Rows: best, mid, and worst features by MSE.

Every sampled feature produces the same basic shape: a scaled ReLU where the positive half tracks a straight line through the origin. The slope printed on each subplot gives the scale factor relative to the ideal ReLU (slope = 1). Qualitatively the responses look similar across best, mid, and worst features, but nine hand-picked samples are not enough to characterize the full picture. The next section measures all 100.

Cross-terms

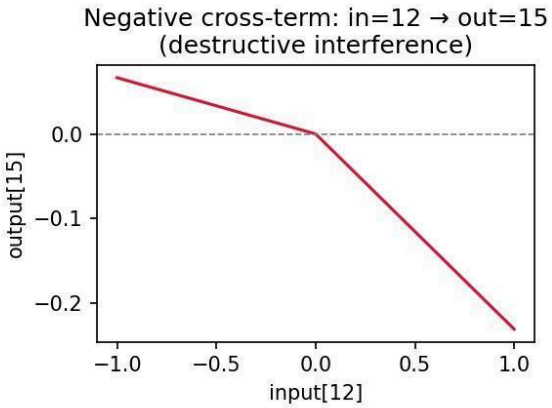
The response plots above only show feature i in -> feature i out (everything else zero). But the trained network routes all inputs through shared neurons, so activating feature i will inevitably affect feature j's output as well. These off-diagonal responses are called cross-terms.

COLAB NOTEBOOK :
https://colab.research.google.com/drive/1yUAGtnTA8rvHFbVvQp0qpXew5qNWbCFr?usp=sharing



Positive cross-terms: best feature (5) activating poorly-reconstructed features. Cooperative interference.

Positive cross-terms are cooperative: when feature i fires, other features get a partial free reconstruction through the same shared neurons. This is not random noise, it is a systematic side effect of the superposition strategy.



Strongest negative cross-term: feature 12 -> feature 15. Activating feature 12 suppresses feature 15's output (destructive interference).

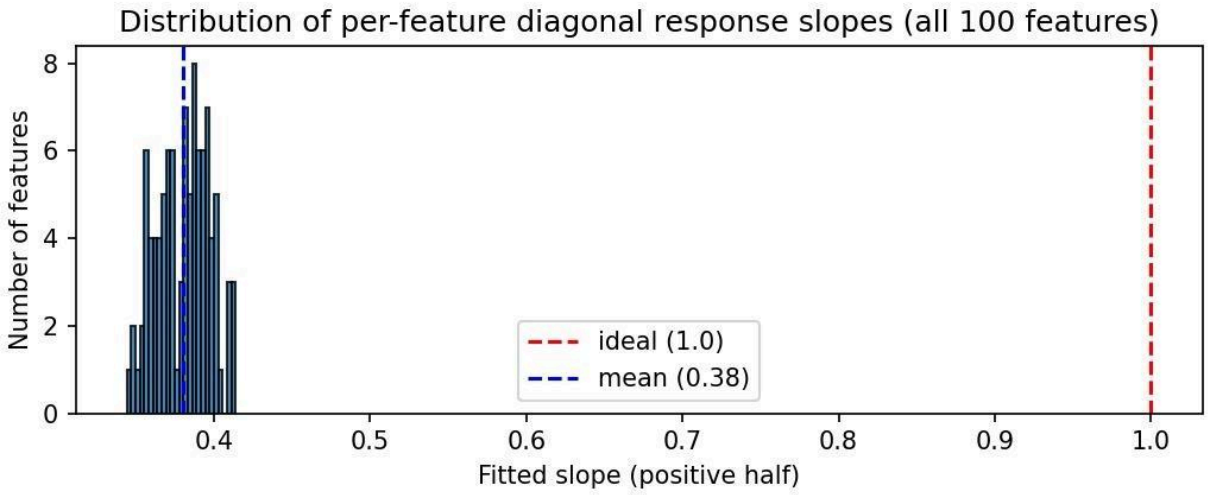
Negative cross-terms are worse: activating feature 12 actively suppresses feature 15's output, with a response of −0.1153 at x = 0.5. This is real interference, not just dilution. The network cannot serve all features perfectly with 10 neurons, so some combinations of active features create destructive interference. **Sparsity (p = 0.02, ~2 features active at a time) keeps this rare enough to be tolerable.**

Slope distribution: all features are treated nearly identically

The qualitative response plots suggested the network might be serving "good" features well and "bad" features poorly. The slope distribution across all 100 features tells a different story.

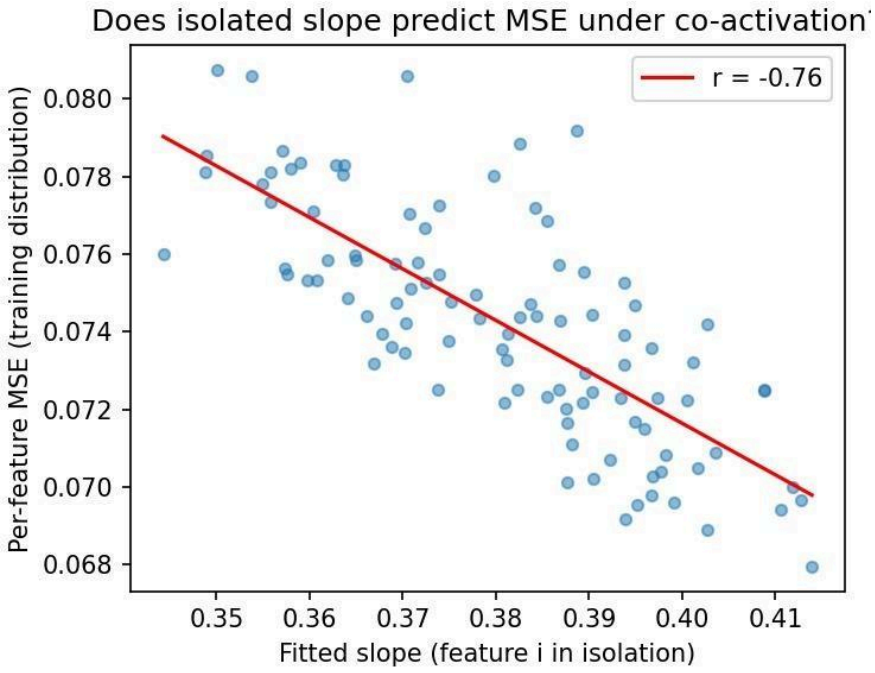
Mean slope: 0.380, std: 0.017. Zero features above 0.8, zero below 0.2. The distribution is remarkably tight. The network learned close to a single shared partial ReLU at ~38% amplitude, applied nearly identically to every feature rather than serving some well and ignoring others. The visual spread in the 3×3 plots was misleading, the "best" and "worst" features differed in their MSE ranking, but their diagonal response slopes were essentially the same.

COLAB NOTEBOOK :
https://colab.research.google.com/drive/1yUAGtnTA8rvHFbVvQp0qpXew5qNWbCFr?usp=sharing



Distribution of per-feature diagonal response slopes across all 100 features. Mean = 0.38; tight cluster well below the ideal slope of 1.0 (red dashed line).

The correlation between isolation slope and MSE under co-activation is r = −0.759. Features with a slightly higher isolated slope also tend to have slightly lower MSE when other features are active alongside them. A feature in a better-conditioned weight direction gets a slightly higher slope and is also slightly less disrupted by cross-terms. Both variations are small in absolute terms (slope std = 0.017, MSE range = 0.0128), so the uniformity is still the main result; this correlation within that narrow range is a secondary finding.



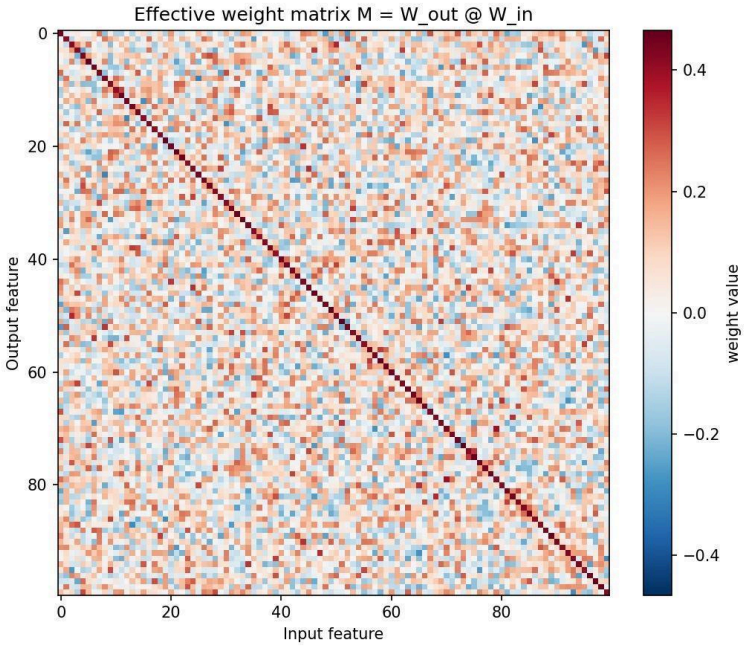
Isolation slope vs per-feature MSE under co-activation (r = −0.759). Features with a slightly higher isolated slope also have slightly lower MSE under training conditions.

COLAB NOTEBOOK :
https://colab.research.google.com/drive/1yUAGtnTA8rvHFbVvQp0qpXew5qNWbCFr?usp=sharing

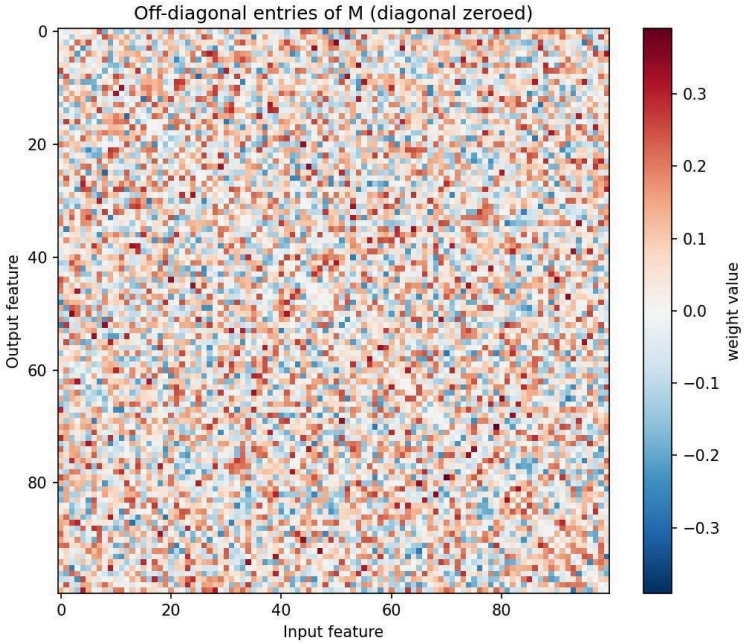
The weight structure

Effective weight matrix M = W_out @ W_in

The full two-layer computation folds into a single linear map M = W_out @ W_in (100×100), followed by the ReLU nonlinearity in the middle. Analyzing M makes the routing structure concrete.



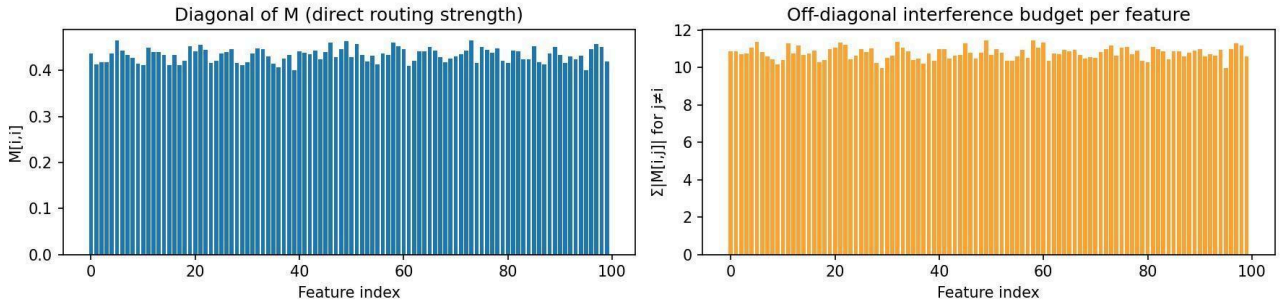
Effective weight matrix M = W_out @ W_in. Strong diagonal visible; off-diagonal entries carry the cross-term interference.



COLAB NOTEBOOK :
https://colab.research.google.com/drive/1yUAGtnTA8rvHFbVvQp0qpXew5qNWbCFr?usp=sharing

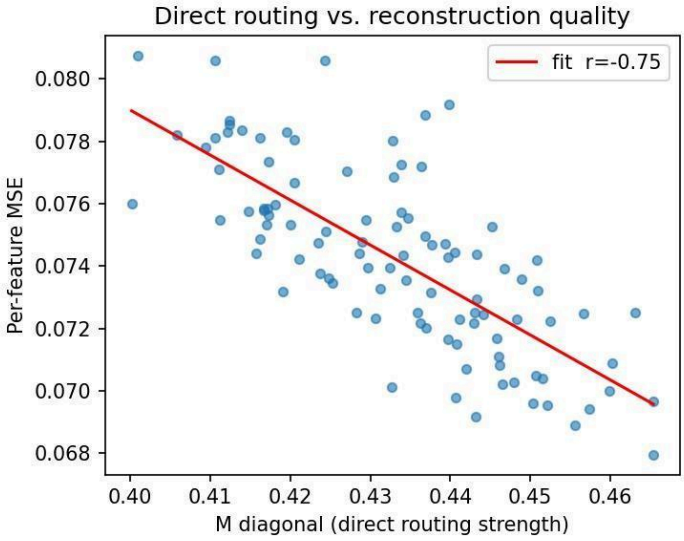
Off-diagonal entries of M (diagonal zeroed). Interference is distributed uniformly with no block structure.

M has rank exactly 10, saturating the bottleneck completely. The diagonal entries M[i,i] represent how directly the feature i gets routed to its own output; the off-diagonal entries are cross-term weights.



Left: diagonal of M (direct routing strength per feature). Right: off-diagonal interference budget per feature (The sum of the absolute values of all non-diagonal entries in row i).

The off-diagonal interference budget is 24.9× larger than the diagonal on average. Under a dense input distribution this would make reconstruction nearly impossible, but with p = 0.02, only ~2 features are active simultaneously, so most of that cross-term weight is rarely triggered at the same time. Sparsity is what makes superposition viable.

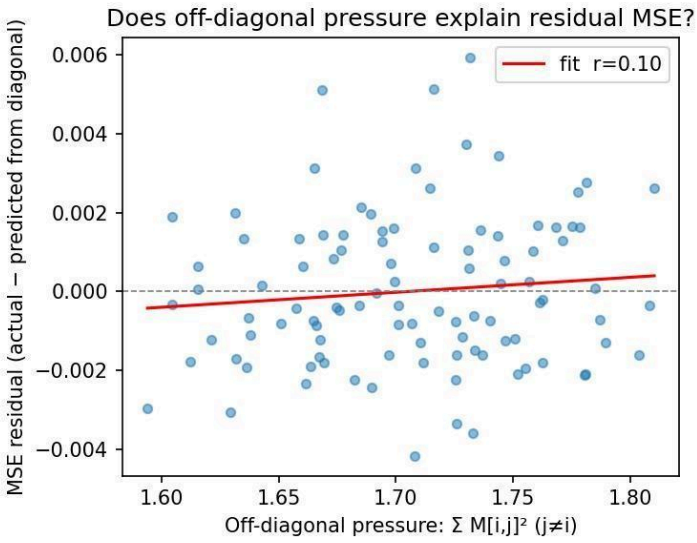


Direct routing M[i,i] vs per-feature MSE (r = −0.750). Stronger direct routing predicts better reconstruction.

What explains the residual MSE?

If the diagonal only explains 56% of MSE variance, what explains the rest? The natural candidate is how much off-diagonal pressure each feature absorbs.

COLAB NOTEBOOK :
https://colab.research.google.com/drive/1yUAGtnTA8rvHFbVvQp0qpXew5qNWbCFr?usp=sharing

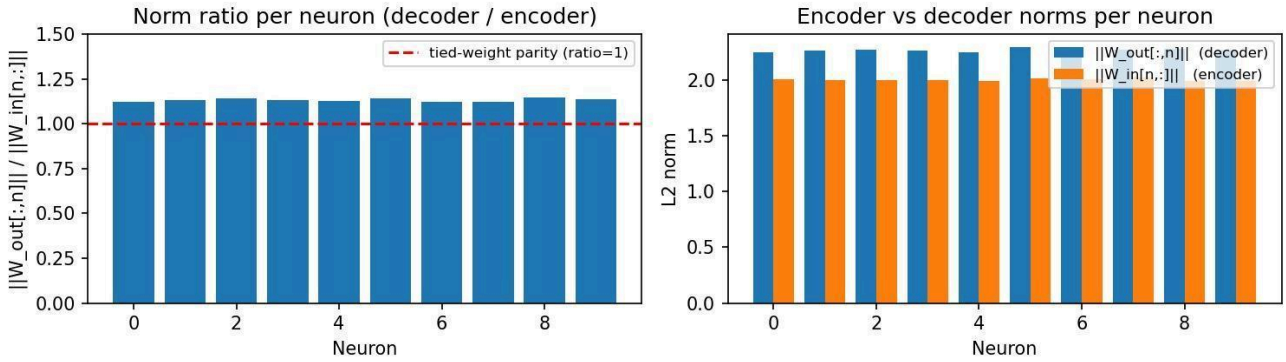


Off-diagonal pressure vs MSE residual. r = 0.103 — essentially no correlation.

r = 0.103 - essentially zero. The off-diagonal pressure does not explain the residual at all. Every feature receives nearly identical off-diagonal interference; there is barely any variance in the predictor to work with. The network arranged its weights so that the interference load is distributed uniformly across all features. The unexplained ~44% variance from the diagonal regression likely comes from conditional gating by the ReLU, where interference depends on **which** features happen to be co-active in a given sample rather than being a fixed linear quantity.

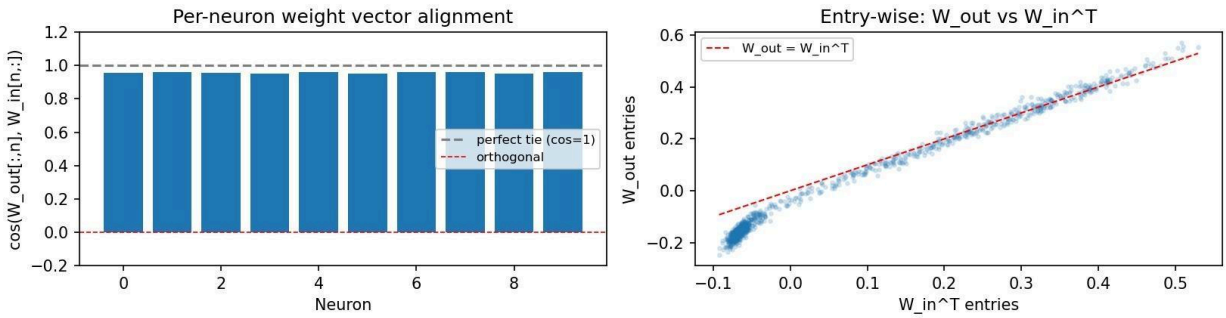
Encoder-decoder norm ratio

The classic superposition model assumes W_out = W_in (transpose) by construction. Here the weights are learned independently. Gradient descent converged toward the tied-weight solution anyway, cosine similarity 0.956, but with the decoder consistently 13% larger in magnitude.



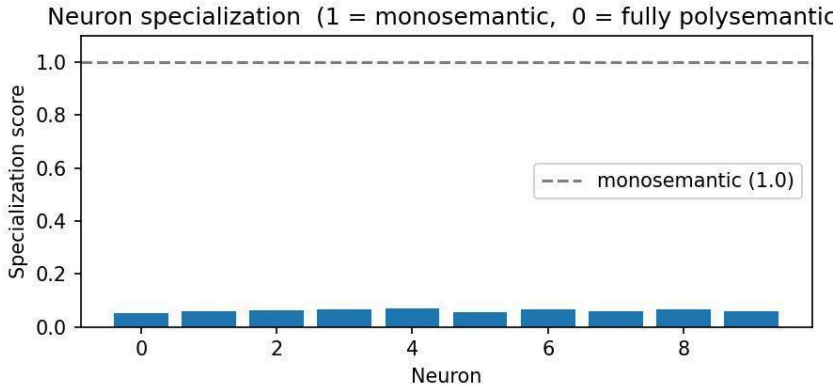
Left: decoder/encoder norm ratio per neuron (~1.13 uniformly). Right: raw L2 norms showing the decoder (blue) consistently larger than the encoder (orange). The ratio is hypothesized to be a ReLU artifact: since ReLU discards negative activations, the decoder compensates.

COLAB NOTEBOOK :
https://colab.research.google.com/drive/1yUAGtnTA8rvHFbVvQp0qpXew5qNWbCFr?usp=sharing



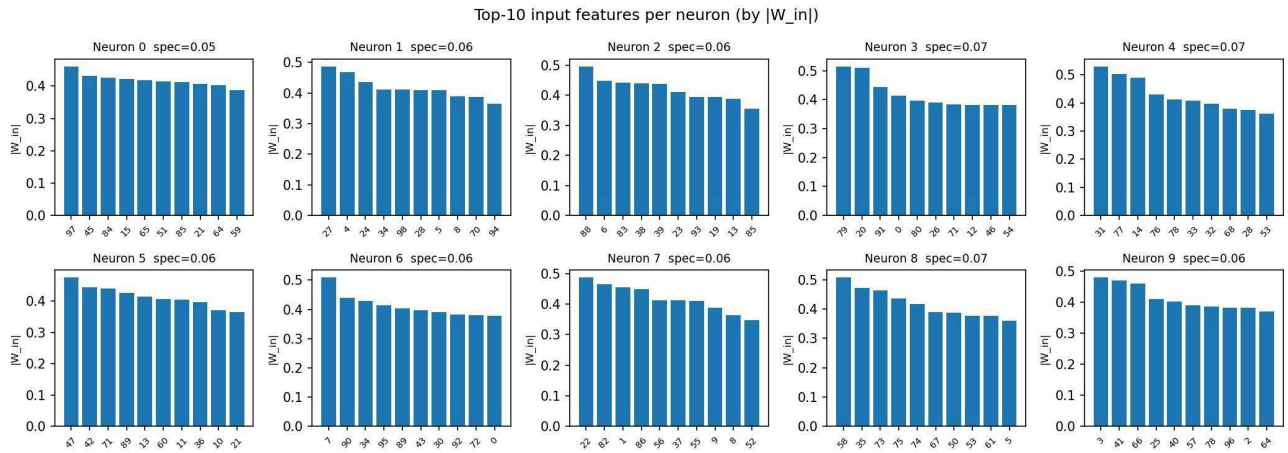
Left: per-neuron cosine similarity between W_out[:,n] and W_in[n,:] — all ~0.95, close to perfect tie (1.0). Right: entry-wise scatter of W_out vs W_in^T showing near-linear alignment with a ~1.13x scale offset (decoder larger).

Neuron specialization: polysemantic by every measure



Neuron specialization scores (1 = monosemantic, 0 = fully polysemantic). All 10 neurons score 0.05–0.08.

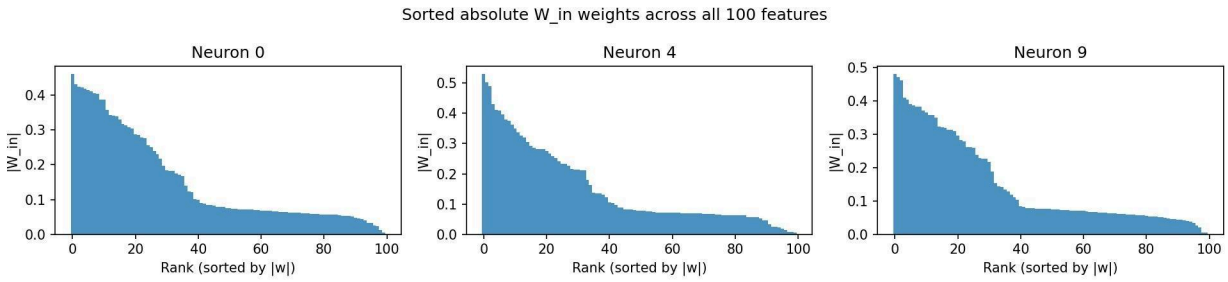
All 10 neurons have specialization scores in the 0.05–0.08 range, well below the monosemantic limit of 1.0 and near the fully uniform limit of ~0.01. No neuron specializes.



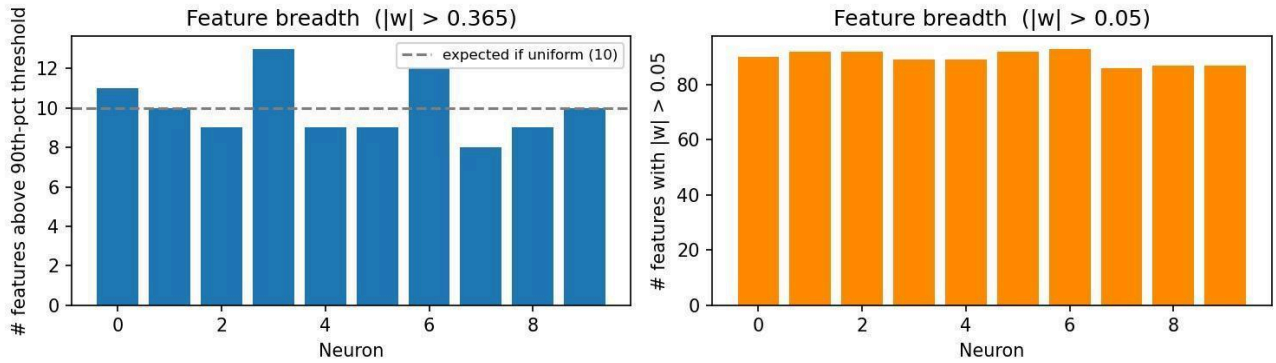
COLAB NOTEBOOK :
https://colab.research.google.com/drive/1yUAGtnTA8rvHFbVvQp0qpXew5qNWbCFr?usp=sharing

Top-10 input features per neuron by absolute(W_in). Weights decline gradually with no dominant feature, typical of polysemantic neurons.

For all neurons: top-1 feature captures only 3% of total weight mass; top-10 combined capture only 27–28%. The weight profile decays almost like a slow power law across all 100 features, with no elbow and no dominant feature.



Sorted absolute W_in weights across all 100 features for neurons 0, 4, and 9. No elbow, no dominant feature — a slow decay across all 100 inputs.



Feature breadth per neuron. Left: features above 90th-pct threshold (~10 each, near the uniform expectation). Right: features with |w| > 0.05 (mean 89.7/100 per neuron, nearly every feature).

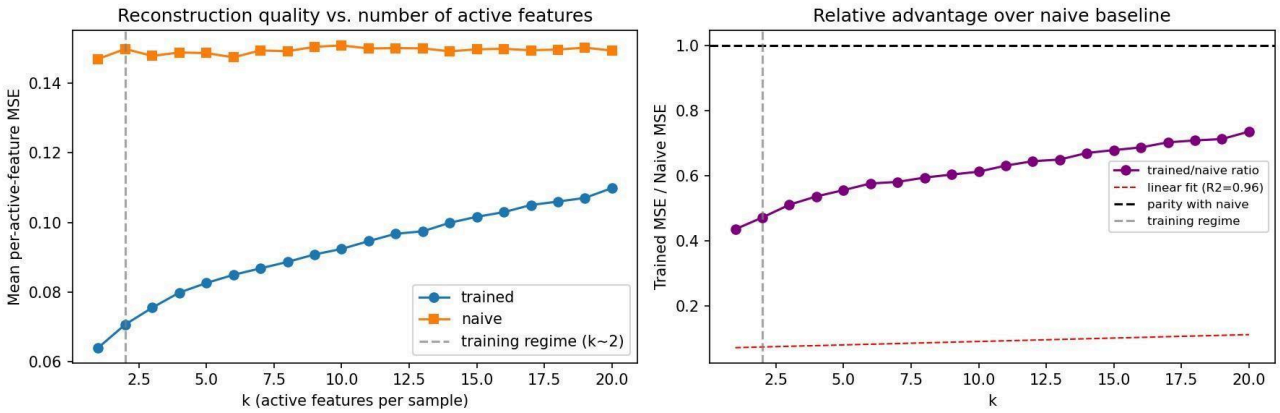
With a low absolute threshold, every neuron maintains non-trivial weight on nearly every feature. The 10 neurons are essentially interchangeable, each with nearly identical coverage of all 100 features.

TASK (part 2)

Performance vs number of active features

The model was trained with p = 0.02 (~2 active features per sample). What happens when we test with more?

COLAB NOTEBOOK :
https://colab.research.google.com/drive/1yUAGtnTA8rvHFbVvQp0qpXew5qNWbCFr?usp=sharing



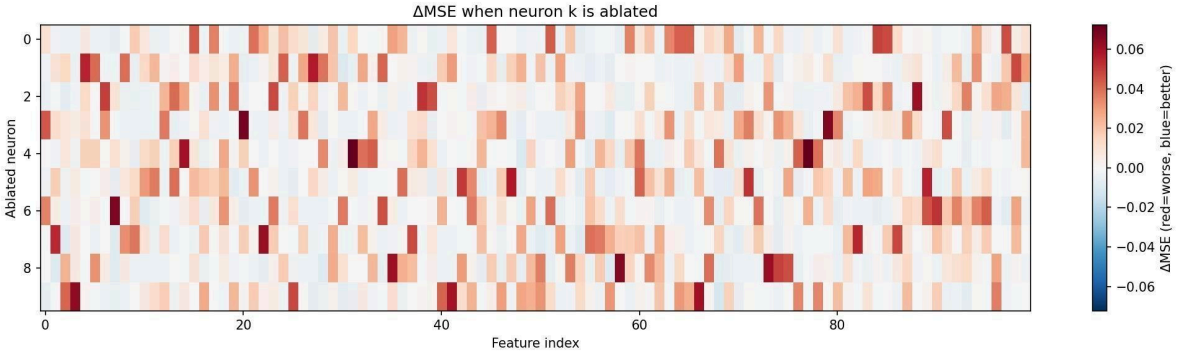
Left: MSE vs k (active features per sample). Right: trained/naive ratio vs k. Linear degradation $R^2 = 0.959$.

The degradation of trained MSE with k is almost perfectly linear (R^2 = 0.959). Each additional co-active feature adds roughly constant noise to all others, the contributions accumulate linearly rather than compounding. This rules out a cascading interference regime.

The trained model never crosses the naive baseline, even at k = 20. At that density it is still 26% better (ratio = 0.736). The naive model also degrades badly at high k: it covers only 10 features, so when k = 20 the other 10 active features all get zero output. The trained model's partial coverage of all 100 stays ahead even at high density.

Ablation analysis: what does each neuron actually control?

Weight magnitudes show which features a neuron is connected to. But the ReLU is in the middle — conditional gating means a neuron can causally affect features it is only weakly connected to. Ablating each neuron (zeroing its W_in row) and measuring the change in per-feature MSE gives the functional picture.

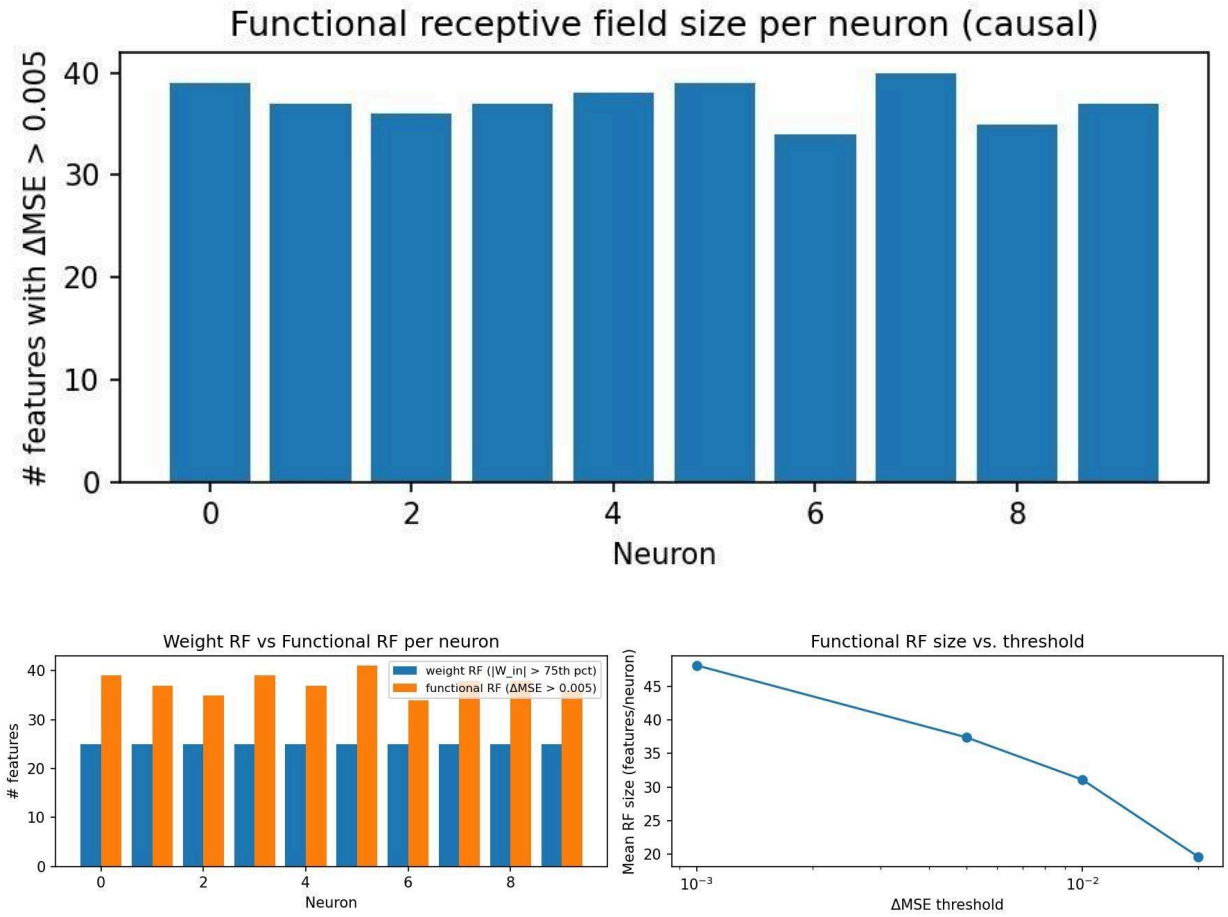


Δ MSE when neuron k is ablated. Red = worse (neuron was helping), blue = better (neuron was interfering). Every neuron damages some features and helps others.

Functional RF vs weight RF

Mean functional RF (ΔMSE > 0.005): 37.4 features/neuron. Mean weight RF (|W_in| above 75th pct): 25.0 features/neuron. Ratio: 1.5x. The ReLU gating extends each neuron's causal reach well beyond what the weight magnitudes alone predict.

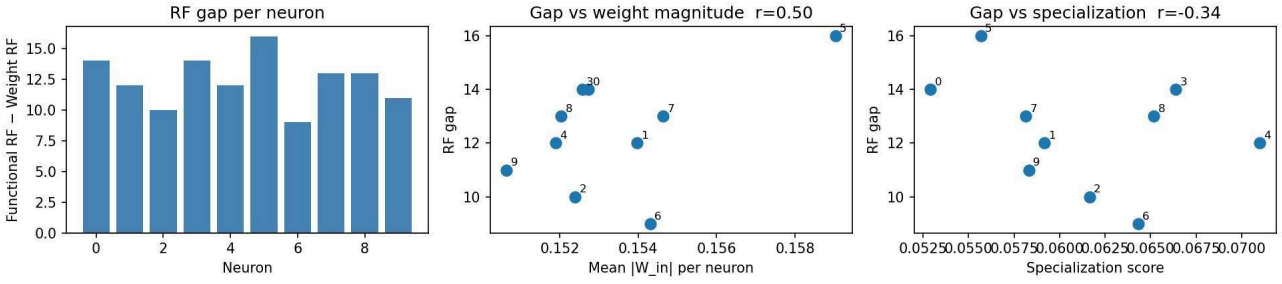
COLAB NOTEBOOK :
https://colab.research.google.com/drive/1yUAGtnTA8rvHFbVvQp0qpXew5qNWbCFr?usp=sharing



Top: functional receptive field size per neuron (features with ΔMSE > 0.005 when ablated), mean 37.4. Bottom left: weight RF (blue, 25.0) vs functional RF (orange) per neuron — functional RF is consistently ~1.5x larger. Bottom right: functional RF size as a function of ΔMSE threshold.

Every neuron hurts some features

Every single neuron, all 10, improves some features when ablated. Neurons 5 and 8 each help 12 features when removed; even the most contained neuron (Neuron 2) helps 4. The interference from superposition is not coming from one or two badly entangled neurons, it is structural and universal. Every neuron is suppressing some features as the price of covering others.

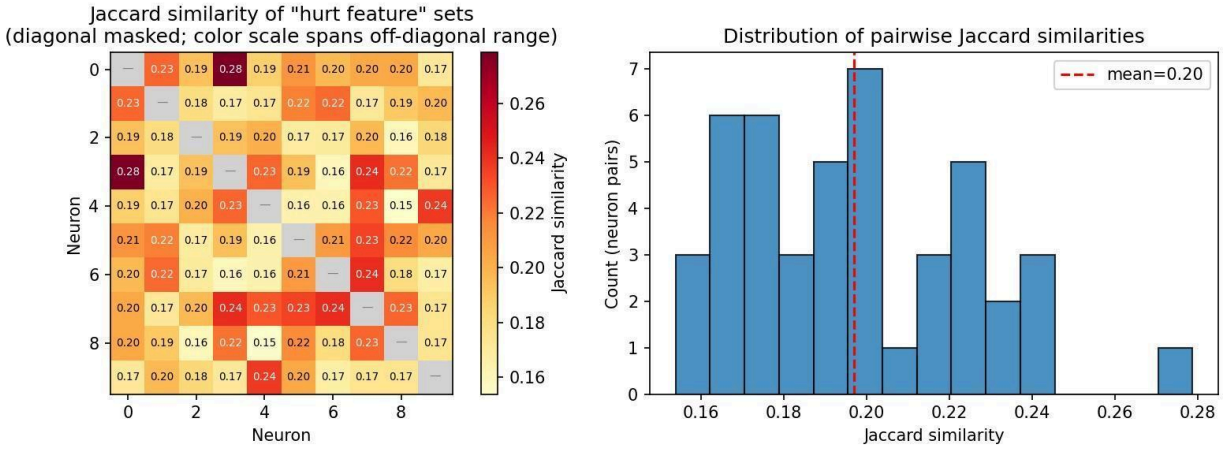


RF gap (functional RF minus weight RF) per neuron. The gap is fairly uniform across neurons (mean 12.4, std 2.0).

Interference footprints are independent across neurons

COLAB NOTEBOOK :
https://colab.research.google.com/drive/1yUAGtnTA8rvHFbVvQp0qpXew5qNWbCFr?usp=sharing

If neurons clustered, two neurons damaging the same features, it would suggest higher-level organization in how the superposition is structured. The Jaccard similarity between each pair of neurons' "hurt-feature" sets tests this.



Jaccard similarity between neuron pairs' hurt-feature sets. Mean = 0.197, std = 0.028. Any two neurons damage largely non-overlapping sets of features.

Mean Jaccard: 0.197, std: 0.028, min: 0.154, max: 0.279. Any two neurons damage largely non-overlapping sets of features, and this holds consistently across all 45 neuron pairs. There is no block structure. The interference pattern looks close to what you would expect from 10 neurons each damaging an independent random subset of features, no hidden cluster structure, just spread-out and roughly independent footprints.

Summary of Work

The 10-neuron network solves the reconstruction problem by committing fully to a distributed strategy:

1) **Every neuron is polysemantic,** encoding all 100 features at low amplitude rather than a few features at high amplitude.
2) **The response for each feature is a scaled ReLU at ~38% amplitude, nearly identical across all 100 features** (slope std = 0.017).
3) **The effective weight matrix M = W_out @ W_in has rank 10 (bottleneck saturated),** with an off-diagonal budget 24.9× larger than the diagonal, but sparsity keeps most of that interference from being triggered simultaneously.
4) **Gradient descent independently discovered the tied-weight solution (W_out ≈ W_in(transpose), cosine 0.956),** but with the decoder consistently 13% larger than the encoder in magnitude.
5) **Performance degrades linearly with the number of active features (R^2 = 0.959),** and the trained model remains 26% better than the naive baseline even at k = 20 active features.
6) **Every neuron suppresses some features as the price of covering others,** but interference footprints are spread independently rather than forming clusters.

Inspiration For Steps Ahead

COLAB NOTEBOOK :
https://colab.research.google.com/drive/1yUAGtnTA8rvHFbVvQp0qpXew5qNWbCFr?usp=sharing

Vary the loss exponent

L4 created strong pressure to serve all features uniformly. Under L2 the model should tolerate larger errors on some features and start specializing. Concrete predictions: slope distribution should widen (currently std = 0.017), some neurons should show higher specialization scores, and the tied-weight cosine similarity should drop below 0.956. Under L6, the uniformity should tighten further.

Test the norm asymmetry against activation type

The 13% scale gap between W_out and W_in (transpose) is uniform across all neurons. The hypothesis is that it is a ReLU artifact: since ReLU discards negative activations, the mean post-ReLU signal is attenuated, and the decoder compensates. Replacing ReLU with absolute value (which discards nothing) should move the ratio toward 1.0.

Find the critical sparsity

The whole mechanism relies on p being small. Somewhere between p = 0.02 and p = 1.0 the accumulated interference should outweigh the benefits of distributed coverage. Running the k-vs-MSE sweep but sweeping p instead (and retraining at each p) would locate where the naive or hybrid strategy starts winning.

Break the uniform interference structure with unequal feature importance

The Jaccard analysis found independent interference footprints because all features have equal loss weight. Retraining with 10x higher loss weight on features 0–9 should concentrate capacity on those features, causing interference footprints to cluster around the high-importance features and the Jaccard matrix to develop block structure. This would confirm that uniform interference is a consequence of equal importance, not a geometric inevitability of the bottleneck.

Vary the bottleneck

At 10 neurons for 100 features, the network chose fully polysemantic with no exceptions. Does this break at 50 neurons? At what compression ratio does even partial monosemanticity become viable under L4? This would map out the phase diagram of the specialization-vs-sharing tradeoff.