What the edit misses in the low runs

> A few fixed-label runs end with the edit removing much less of `difference` than the rest, and nothing that training sees sets them apart. A run trained again from the same labels, seed, and batches matches a low run in loss and in alignment°, and still removes the op well. The difference is in what the later blocks read. In every run, the evidence that an example fits `difference` lies partly on the anchored axis and partly off it. In a low run, the blocks that read this evidence take more of it from the off-axis part, which the edit never touches.

Alignment. How far a state points along e_1....

The *fixed-labels*° experiment found that holding one label draw for a run leaves a few runs in eight well below the rest in removal°. Most of them never reach the level of the others, their label sets look like any other, and retraining moves which runs end low (E2 to E5 of that report). This page scores the stored checkpoints of all 48 runs of that experiment and of the *fixed-confirm*° experiment, with no new training. It asks what a low run does differently with the op.

fixed-labels. With one label draw held for the run, the anchor settles at every rate, but a few runs in eight end well below the rest in removal, and which runs do so moves when a run is trained again with slightly different arithmetic.

Removal. How much of `difference` the full e_1 edit removes, beyond what the same edit removes along a typical other axis, as the fixed-labels experiment scores it....

fixed-confirm. `τ_ℓ fixed`, with the per-visit label draw, keeps the anchor on all eight runs at seeds 712 to 719 with removal near 0.9, and is the recipe of record for D2.2.

Observations

- Twins that split (E1): two label sets were each trained twice from the same seed and batches, and each pair ended with one run low and one not. Within a pair, validation loss and α° at the example answers stay within a hundredth of each other for the whole run. Removal° parts ways early in one pair and late in the other. By the first checkpoint, the weights of a twin pair are already most of the way to unrelated. So the last bits of the arithmetic make two different runs that happen to share a function.

  α. Alignment with e_1...

  Removal. How much of `difference` the full e_1 edit removes, beyond what the same edit removes along a typical other axis, as the fixed-labels experiment scores it....

- What survives the edit (E2): after the edit, the model still gets right the contexts whose three examples all fit `difference`. Unexpectedly, the low runs keep several times as many of them. In every run, the evidence that an example fits has two parts, one on e_1 and one along a single direction off it, the direction a non-fitting answer has, and a linear probe reads either part about as well.
- Where the edit misses (E3): before any baseline, the e_1 edit leaves a fifth of the way in the low runs, against a twentieth in the rest. Removing e_1 at the three example answers alone removes as much as the full edit. In half the low runs, block 4 also reads the answers, through their off-axis content. In the other half, the usual reader, block 3, takes part of the evidence from off the axis. Clearing the answers or the other example positions on top of the full edit lifts most low runs toward the rest, and which of the two does so differs by run.
- Peak and slide (E4): the low runs peak lower than the rest, and give back a little more after the peak while the anti weight anneals. The off-axis evidence is in place early in every run, low or not. The two groups do not separate in removal until after epoch 200.

Scope and method

This is an exploratory re-analysis of 48 stored runs, with no preregistration and no gate. They are the 24 fixed-label runs of the *fixed-labels*° experiment (three rates at seeds 712 to 719), its 16 crossed runs (the 1/50 sets of four seeds, each trained at all four), and the eight per-visit runs of the fixed confirmation. All use the recipe of record, `τ_ℓ fixed`, trained for 400 epochs.

***fixed-labels*.** With one label draw held for the run, the anchor settles at every rate, but a few runs in eight end well below the rest in removal, and which runs do so moves when a run is trained again with slightly different arithmetic.

Four of the crossed runs pair a label set with its own seed. Each repeats a *fixed-labels* run from the same initialization, on the same batches, with the same labels. The one difference is the card: the *fixed-labels* runs trained on L4s and the crossed runs on L40S cards. XLA, the compiler that turns the model into GPU code, compiles a different program for each card model, and a sum taken in a different order differs in its last bits. So the two runs of a pair see the same gradients to within rounding, and part by a little on the first step. We call them twins. Two of the four pairs ended with one run low and one not.

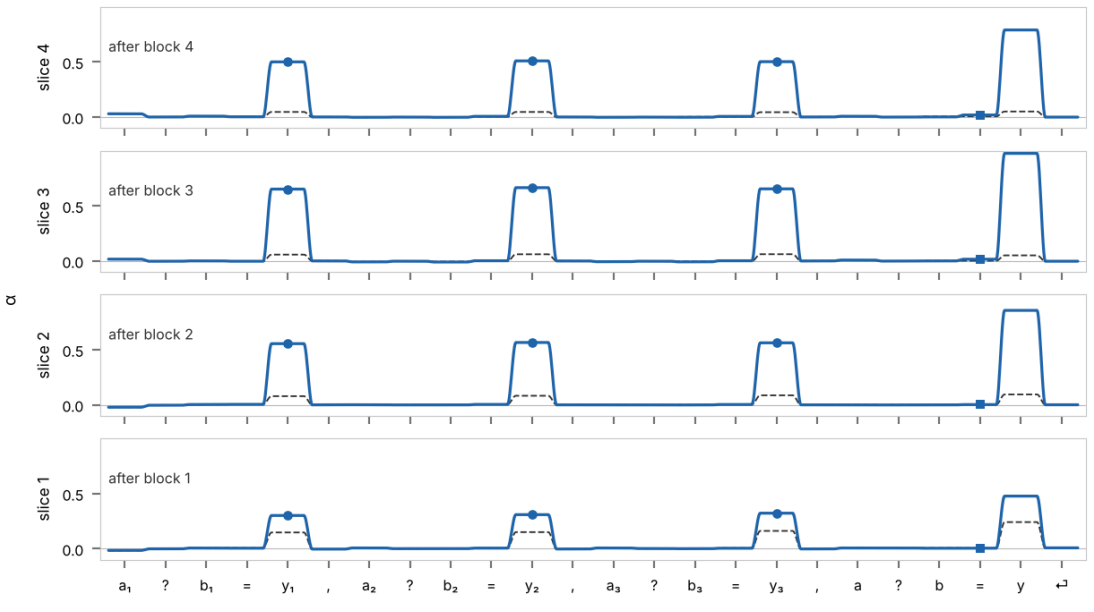
A run is low when its removal° ends below 0.8, under the lowest per-visit run, as the *fixed-labels* report drew the line. That makes 10 low runs: the five of the *fixed-labels* report, and five of its crossed runs. No per-visit run is low. Removal spreads by about 0.07 over the eight per-visit seeds, so a gap of 0.2 between a low run and its twin is well outside the seed range.

Removal. How much of `difference` the full e_1 edit removes, beyond what the same edit removes along a typical other axis, as the fixed-labels experiment scores it....

Every pass uses the 14,000 held-out contexts of *ex-2.2.21*°, 2,000 per op, and the probes score on half of them. Restricted edits (one slice, or one set of positions) are scored without the median-over-axes baseline, which would need 63 more passes per edit. The baseline takes about a tenth off the full edit in every run. So a restricted edit can be compared with the full edit scored the same way, and only loosely with removal. The slide and peak of E4 use the kept checkpoints of the *fixed-labels* experiment and the fixed confirmation, scored there.

***ex-2.2.21*.** The in-context grammar with three examples and its held-out set of 14,000 contexts, 2,000 per op.

The measurements



Where the probes read. *Alignment with e_1 (α) by position in a held-out context, one panel per slice after a block, on one run (fixed 1/50, retrained at 715). The* `difference` *contexts are solid and the mean over the other ops is dashed. Circles mark the three example answers y_1 to y_3, where the anchor pulls and the fits probe reads the state (with e_1 zeroed, or e_1 alone). A square marks the query* `=`*, where the query probe reads. Between one panel and the next sits one block: attention, then its MLP.*

Removal

How much of `difference` the full e_1 edit removes, beyond what the same edit removes along a typical other axis, as the fixed-labels experiment scores it. The full edit removes e_1 at every slice and position. Its drop on `difference` is taken as a share of the way from the clean score to the target null, less the median drop under the same edit along each of the other 63 axes. Higher is better.

Edit share

How much of `difference` an edit removes, with no baseline: its drop on `difference` as a share of the way from the clean score to the target null. The edit is the full edit, one slice of it, or an edit that removes every direction but e_1 (or the whole state, with no renormalization) at a set of positions.

Fits probe

Whether an example fits `difference`, decoded from the state at its answer. An example fits when its answer is one `difference` gives for its operands. The others carry replacement op noise. The probe is a ridge (regularized linear) probe on that state with the e_1 coordinate zeroed, fitted on half the held-out contexts of every op and scored on the rest by AUC: the chance that it ranks a fitting example above one that does not fit, so 0.5 is chance and 1 is perfect. It is shown beside the AUC of the e_1 coordinate alone. The figure above shows where the probe reads. E2 also scores one fixed off-axis direction, the one between the mean states of fitting and non-fitting examples, on the zeroed state and on the zeroed state renormalized to unit length.

Query probe

Whether the op of a context is `difference`: the same probe on the state at the query `=`.

Edited match

How often the edited model still gets the query right: the expected exact match on the query answer of a held-out `difference` context after the full edit, split by how many examples of the context fit `difference`. Lower means more of the op removed.

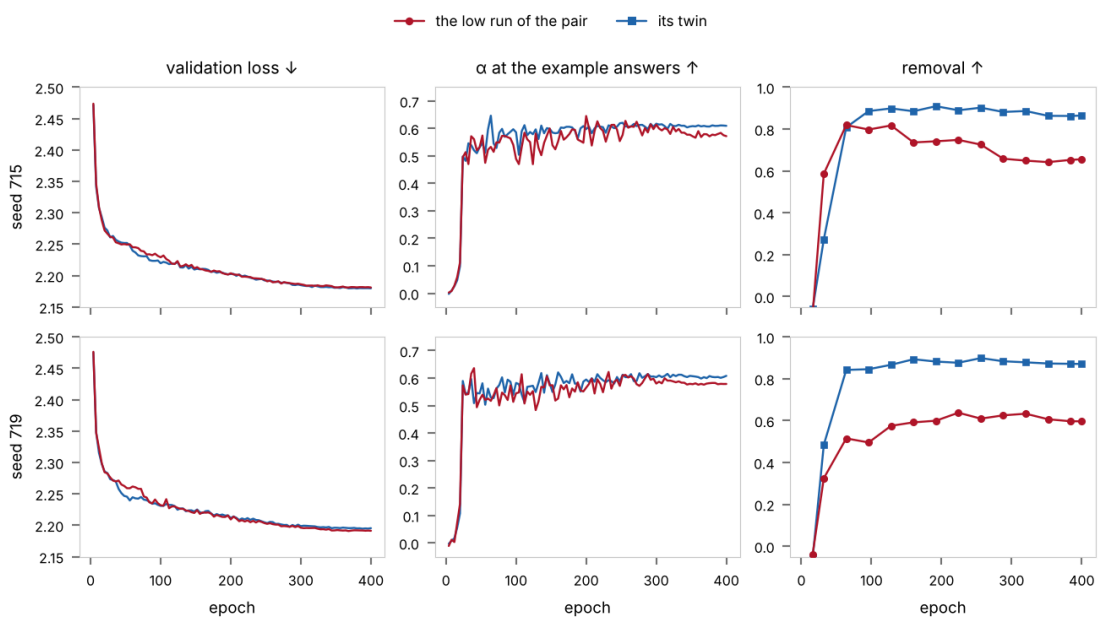
α

Alignment with e_1: the e_1 component of the unit-norm state at the example answers of the held-out `difference` contexts, at slice 3.

Twins that split (E1)

Does a low run look different to training? The twins give the cleanest test. Two of the four pairs ended one low and one not, at seeds 715 and 719. If a low run went a different way on the task or on the anchor, its loss or its α° would show it somewhere along the run.

α. Alignment with e_1...



The two twin pairs that split, along training. *Each row is one label set at one seed, trained twice: once in the fixed-labels experiment and once in its crossed retraining. The two runs differ only in the last bits of the arithmetic. The low run of the pair is in red, the other in blue.* ***Left:*** *validation loss.* ***Middle:*** *α at the example answers of the* `difference` *probe contexts, at slice 3.* ***Right:*** *removal under the full edit at the kept checkpoints, net of the median over the other 63 axes.*

They do not. In both pairs, the validation loss and α at the example answers stay close for the whole run, while removal° under the full edit separates by about 0.2. The loss curves do part a little in places. Averaged over any 40 epochs, the two runs of a split pair differ by at most 0.014, sometimes in favor of the low run and sometimes against it. But the two pairs that did not split differ by as much (0.013), so these gaps are the ordinary difference between any two runs. Over the last hundred epochs, α in the low run ends a few hundredths under its twin, less than the spread across seeds. At seed 719 the two removal curves part by epoch 32 and never meet again. At seed 715 they run together until about epoch 100, and then the low run slides while its twin keeps climbing.

Removal. How much of `difference` the full e_1 edit removes, beyond what the same edit removes along a typical other axis, as the fixed-labels experiment scores it....

run	val loss ↓	α, answers	removal ↑	full edit	e_1, slice 2	e_1, slice 3	all but e_1, answers	full, then answers cleared	full, then others cleared
fixed 1/50 at 715 (low)	2.181	0.61	0.66	0.74	0.82	0.04	−0.15	0.88	0.79
fixed 1/50, retrained at 715	2.180	0.66	0.86	0.94	0.96	0.02	−0.18	0.87	0.92
fixed 1/50 at 719 (low)	2.191	0.62	0.60	0.69	0.72	0.03	−0.22	0.62	0.84
fixed 1/50, retrained at 719	2.195	0.65	0.87	0.94	0.95	0.06	−0.17	0.87	0.93

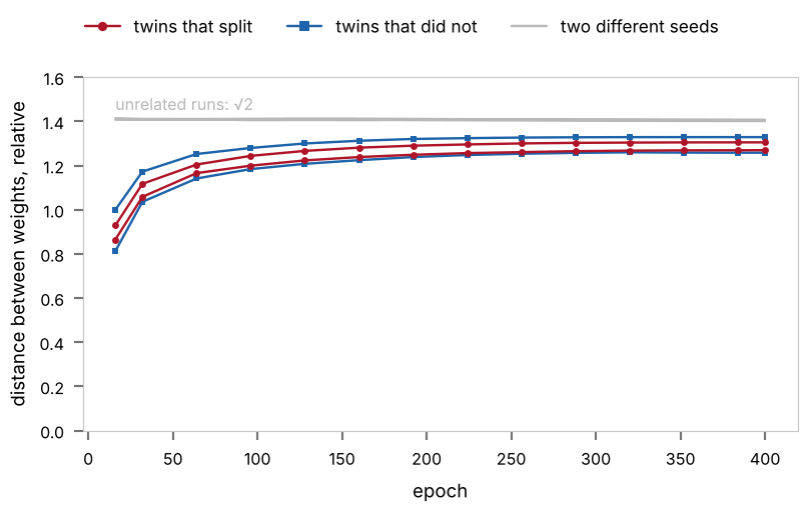
The two split pairs at epoch 400. *Validation loss and α at the example answers (slice 3) are what training sees, and the rest are edits. Removal is net of the median over the other axes. The other columns are shares of the way with no baseline: under the e_1 edit at every slice, at slice 2 or 3 alone, with every direction but e_1 removed at the answers after block 2, and under the full edit with the whole state then removed at the answers, or at the other example positions, after block 2.*

So the low state costs nothing on the task and nothing on the anchor term. That fits what the *fixed-labels*° experiment found about label sets and seeds: training is indifferent to which way a run goes on this. What differs between twins has to be something the edit sees and the loss does not. The rest of the table narrows it a little. In both pairs, the e_1 edit at slice 2 alone already misses more in the low run, so the twins part at the reader at block 3. (The *block1-edit*° experiment found the reader at various later blocks on other seeds, and E3 covers those.) Where the reader still finds the evidence differs by pair. At seed 719, clearing the other example positions on top of the full edit brings the low run most of the way to its twin, and clearing the answers does nothing. At seed 715 it is the other way round. E3 follows this up across all the low runs.

fixed-labels. With one label draw held for the run, the anchor settles at every rate, but a few runs in eight end well below the rest in removal, and which runs do so moves when a run is trained again with slightly different arithmetic.

block1-edit. Block 1 writes the evidence for the op onto e_1 at the example answers, one later block reads it from the query `=` through attention, and an edit removes the op when it sits between the two.

How can the last bits of the arithmetic decide this? Twins part early and keep parting, so they are two different runs rather than one run with a small change at the end. The distance figure measures how far apart the weights of a pair are, as a share of their size. On that scale, two unrelated runs sit at $\sqrt{2} \approx 1.41$, and the *fixed-labels* runs at different seeds sit at 1.40 (1.40 to 1.41) at every checkpoint. By epoch 16 the twins are already 0.90 (0.81 to 1.00) apart, most of the way to unrelated, and the gap keeps widening to 1.29 (1.26 to 1.33) at epoch 400.



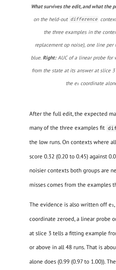
How far apart the twins drift. *The distance between the weights of the two runs of a twin pair, as a share of their norm, at each kept checkpoint and at the end. One line per pair: the pairs that split in red (circles, seeds 715 and 719) and the pairs that did not in blue (squares, seeds 712 and 718). The grey lines are the same distance between the fixed-labels runs at two different seeds.*

So by the end, a twin pair is nearly as far apart in weight space as two seeds, while its loss and α agree to a hundredth. The pairs that split are no further apart than the pairs that did not. Twins are different runs that share a function, so a split needs no rare tipping point for a last bit to cross. Which side of the line each run lands on is drawn afresh, with odds that these seeds put near even. Training is known to be sensitive to any perturbation at all. What is new here is which quantity the perturbation changes.

What survives the edit (E2)

Which contexts does the edited model still get right, and what does the model know about `difference` that is not on e_7? In the *block1-edit*[1] account, block 1 writes the evidence that an example fits `difference` onto e_7 at the answer, and a later block reads it. The edit removes the op by clearing that coordinate. If the evidence also had a part off the axis, the reader could take it from there, and the edit would miss that part.

[1] **block1-edit**: Block 1 writes the evidence for the op onto e_7 at the answer, and a later block reads it from the same place. The edit clears that coordinate, and a edit- to-edit account is the part of the evidence that is not on e_7.

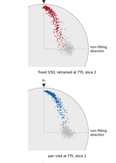


What survives the edit, and what the probes find. Left: *expected exact match on the held-out* `difference` *contexts under the full edit, by how many of the three examples in the context fit* `difference`*. (the rest carry multiplicative noise), one line per run. the low runs in red, the others in blue. **Right:** AUC of a linear probe for whether an example fits* `difference`*, from the state at its answer at slice 3 with e_7 zeroed (x), against the AUC of the e_7 coordinate alone (y), one mark per run.*

After the full edit, the expected match rises in every run with how many of the three examples fit `difference`, and far more steeply in the low runs. On contexts where all three fit, the edited low runs still score 0.32 (0.20 to 0.45) against 0.06 (0.02 to 0.14) for the rest. On the noisier contexts both groups are near the floor. So what the edit misses comes from the examples that fit.

The evidence is also written off e_7 in every run. With the e_7 coordinate zeroed, a linear probe on the state at an example answer at slice 3 tells a fitting example from one that does not fit at AUC 0.98 or above in all 48 runs. That is about as well as the e_7 coordinate alone does (0.99 (0.97 to 1.00)). The low runs sit among the rest on both. The op at the query `=` is off-axis too, at AUC 0.88 (0.65 to 0.94), as *block1-edit* found: the anchor never held it there.

How many directions does the off-axis evidence take? Few. Here the probe is given only the top principal components of the off-axis state. At slice 2 it needs a median of 2 components to reach AUC 0.95, though a handful of runs need 8 or more, and one never gets there. At slice 3, one component is enough in 46 of 48 runs. A single fixed direction, the one between the mean states of fitting and non-fitting examples, scores 0.99 (0.94 to 1.00) at slice 2 and 0.99 (0.95 to 1.00) at slice 3. By squared length, that mean difference lies 0.75 (0.64 to 0.83) along e_7 at slice 2 and 0.58 (0.55 to 0.66) at slice 3, with the rest along one off-axis direction.



How many directions the off-axis evidence takes. *AUC of the fits probe given only the top principal components of the state at the example answers with e_7 zeroed, by number of components. One line per run: the low runs in red, the rest in blue. **Left:** slice 2. **Right:** slice 3.*

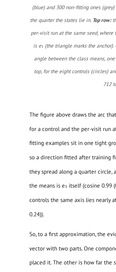
What is that off-axis direction? Two checks say what it is and that it is real. First, the states are unit vectors, so a large e_7 coordinate leaves less length for everything else, and a probe on the zeroed state could be reading that shadow of e_7 rather than a direction. Renormalizing the zeroed states removes the shadow. The direction between the renormalized class means still scores AUC 0.98 (0.94 to 1.00) at slice 3, where the reader reads, and 0.93 (0.78 to 0.98) at slice 2, so the off-axis evidence is a direction the state has. Second, along that direction a non-fitting example sits at −0.86 (−0.95 to −0.75) and a fitting one at −0.21 (−0.35 to −0.08) at slice 3: the direction is the one an answer state has when its example does not fit, and a fitting example has moved away from it, toward e_7. Among the fitting examples the e_7 coordinate and this coordinate move together ($r = 0.66$ (0.48 to 0.86)), so examples do not divide into some that fit in one respect, written on e_7, and others that fit in another, written elsewhere. They lie along one arc, from the non-fitting state to e_7, and differ in how far along it they sit.



The arc from the non-fitting direction to e_7. *States at the example answers in the plane spanned by the direction a non-fitting answer has (horizontal) and e_7 (vertical), as in the next figure: 400 fitting examples (red for the low run, blue for the others) and 400 non-fitting ones (grey) per panel. **Rows:** the low run at seed 715, its twin, and the per-visit run at the same seed. **Columns:** slice 2, slice 3. The states are unit vectors, so a state that lies in this plane sits on the rim of the grey unit disc, shown cropped to the quarter the states lie in; one inside has part of its length in other directions. The triangle marks e_7, the anchor.*

The figure draws that arc for three runs at seed 715: the low run, its twin, and the per-visit run. Each answer state is a unit vector, so in the plane of e_7 and the non-fitting direction a state that lies in the plane sits on the unit circle, and one that sits inside the circle has part of its length in other directions. The non-fitting examples gather on the right, where e_7 is near zero: on the circle at slice 3, and inside it at slice 2. The plane is a projection, so "arc" is a simplification: two non-fitting examples that land on the same point here can be far apart in the other directions, as the ones at slice 2 are, and the figure says nothing about whether an example that fits `lighten` and one that fits `mix` sit near each other. What it does say is that every fitting example has left the non-fitting direction by about as much as it has gained on e_7. The fitting examples spread up the circle toward $e_7 = 1$, further along at slice 3 than at slice 2, and at slice 3 the ones near the top sit at e_7 close to 1. The three runs have the same shape; the fitting examples of the low run sit a little further from e_7 at slice 3, which is the small gap in the table of E1. So the writer gives the readers much the same picture in a low run, its twin, and a per-visit run, which is the point of E1 seen from inside.

A model trained without the anchor holds the same arc, found after training from the mean states of the two kinds of example, but a short one: in the eight controls at seeds 712 to 719 the two means lie a median of 21° apart at slice 3 (9° to 35°), where the per-visit runs put them 83° apart (arc-vs-controls probe). So the anchor stretches the arc and names one end of it in advance, which makes the arc quicker to read; it is there to be found without it.



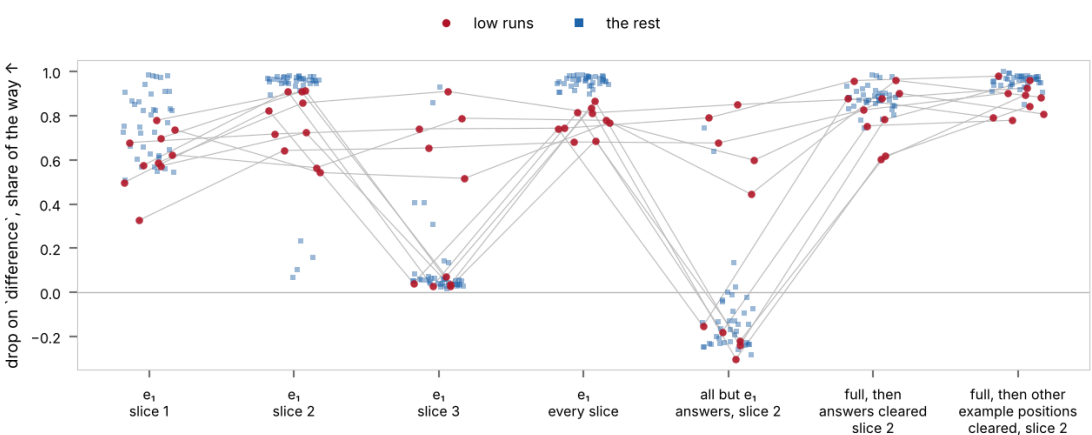
The arc without the anchor. *States at the example answers in the plane of the two class means, with no chosen axis: the direction of the non-fitting mean (horizontal) and the part of the fitting mean at right angles to it (vertical), a tick on the rim marks the fitting mean itself. 300 fitting examples (blue) and 300 non-fitting ones (grey) per panel, on the unit disc, cropped to the quarter the states lie in. **Top row:** the control at seed 715. **Middle row:** the per-visit run at the same seed, where the vertical axis found from the means is e_7 (the brace marks the arc). **Columns:** slice 2, slice 3. **Bottom:** the angle between the class means, one faded mark per run and the mean on top, for the eight controls (circles) and the eight per-visit runs (squares) at 712 to 719.*

The figure above draws the arc that way, from the class means alone, for a control and the per-visit run at seed 715. In the control the fitting examples sit in one tight group just off the non-fitting group, so a direction fitted after training finds both ends. In the per-visit run they spread along a quarter circle, and the end the plane finds from the means is e_7 itself (cosine 0.99 (0.98 to 1.00)), where in the controls the same axis lies nearly at right angles to e_7 (0.04 (−0.19 to 0.24)).

So, to a first approximation, the evidence that an example fits is one vector with two parts. One component lies along e_7, where the anchor placed it. The other is how far the state has left the direction a non-fitting answer has, which the loss made on its own and the anchor never touched. Both parts are there in every run, low or not, and nothing in the end state of the writer (block 1) sets the low runs apart. What has to differ is how much the readers take from each part.

Where the edit misses (E3)

Where does the reader take the rest of the evidence from in a low run? The edits here change the full edit in two ways. Some narrow it to one slice. Others go further than e_1 at one set of positions after block 2. One of these removes every direction but e_1 at the example answers, so the next block sees the answers through e_1 alone. The other two apply the full edit and then remove the whole state at the answers or at the other example positions, so whatever the full edit left there is gone too. Where that last kind lifts a low run toward the normal runs under the same edit, the leftover was reaching the reader from those positions. Clearing a whole state is a diagnostic, too costly to use as an intervention: it costs the other ops up to 0.63 in expected match, against 0.10 at most under the full edit.



Where the edit reaches the evidence, by run. *The drop on* `difference` *under each edit, as a share of the way from the clean score to the target null, before netting against the other axes. One mark per run: the low runs in red and the rest in blue, with a thin grey line joining the edits of each low run. The first four edits remove e_1 at the slice named, at every position. The fifth removes every direction but e_1 at the example answers after block 2. The last two apply the full edit and then remove the whole state (every direction, no renormalization) at the example answers, or at the other example positions, after block 2.*

Before any baseline, the full edit misses more in the low runs. It takes 0.77 (0.68 to 0.87) of the way against 0.95 (0.90 to 0.98) in the rest, so the baseline is not what sets the groups apart. Removing e_1 at the three example answers alone removes as much as the full edit in every run, to within 0.02. So the e_1 that the full edit takes off the other positions makes no difference. No single slice accounts for the gap either. In most runs, low or not, the e_1 edit at slice 2 alone removes nearly as much as the full edit. That is the reader at block 3 of the *block1-edit*° account. A few runs, all at seed 712, read at block 2 instead, and the edit at slice 1 removes the op in them.

***block1-edit*.** Block 1 writes the evidence for the op onto e_1 at the example answers, one later block reads it from the query `=` through attention, and an edit removes the op when it sits between the two.

The low runs come in two shapes. In 5 of the 10, the e_1 edit at slice 3 alone also removes much of the op, so block 4 reads it from the answers as well. Keeping only e_1 at the answers after block 2 costs these runs 0.67 (0.44 to 0.85) of the way, so that reader uses the off-axis content of the answer states. The other 5 look like normal runs in where the op is read (slice 2, block 3) and under the all-but-e_1 edit, which costs them nothing: block 3 takes e_1 when it is there. The clearing edits below show where both shapes get their leftover.

run	removal ↑	full edit	e_1, slice 3	all but e_1, answers	full, then answers cleared	full, then others cleared
set of 718 at 719	0.60	0.68	0.65	0.68	0.83	0.90
fixed 1/50 at 719	0.60	0.69	0.03	−0.22	0.62	0.84
fixed 1/50 at 715	0.66	0.74	0.04	−0.15	0.88	0.79
fixed 1/10 at 717	0.66	0.74	0.74	0.79	0.96	0.98
set of 712 at 718	0.68	0.77	0.52	0.60	0.90	0.81
set of 719 at 718	0.70	0.78	0.79	0.44	0.96	0.88
fixed 1/10 at 718	0.72	0.81	0.91	0.85	0.88	0.93
set of 712 at 715	0.74	0.82	0.03	−0.18	0.75	0.78
fixed 1/10 at 719	0.75	0.83	0.07	−0.30	0.60	0.89
set of 718 at 715	0.79	0.87	0.03	−0.24	0.78	0.96
the normal runs	0.87 (0.82 to 0.91)	0.95 (0.90 to 0.98)	0.12 (0.02 to 0.93)	−0.11 (−0.28 to 0.75)	0.88 (0.74 to 0.98)	0.96 (0.85 to 1.00)

***The low runs under the edits of this section, at epoch 400**, lowest removal first, with the mean and range of the 38 normal runs in the last row. The columns are as in the table of E1. A run is counted as lifted by a clearing when its share rises by 0.1 or more over the full edit; no normal run rises by more than 0.08 under either.*

In the twins, clearing the answers on top of the full edit brings the low run at seed 715 level with its twin, and clearing the other example positions does not. At seed 719 it is the other way round. Across the ten low runs, one clearing or the other lifts the full edit by a tenth or more in 7 (5 by the answers, 5 by the other positions, and 3 by both). In the normal runs neither clearing lifts it by more than 0.08. The 3 low runs that neither clearing lifts are the 3 with the smallest shortfall. So the leftover sits in the off-axis content of the example positions, at the answers or at the others depending on the run. The e_1 edit leaves it because it was never on e_1.

A reader after block 3 does not make a run low on its own: 2 normal runs have one too, and remove the op well. What the low runs share is that their readers take enough of the evidence from off the axis for the e_1 edit to leave about a fifth of the op in place.

measurement	low (10 runs)	the rest (38 runs)
removal	0.69 (0.60 to 0.79)	0.87 (0.82 to 0.91)
full edit, share of the way	0.77 (0.68 to 0.87)	0.95 (0.90 to 0.98)
e_1 edit at slice 2 alone	0.76 (0.54 to 0.92)	0.87 (0.07 to 0.98)
e_1 edit at slice 3 alone	0.38 (0.03 to 0.91)	0.12 (0.02 to 0.93)
all but e_1 removed at the answers, slice 2	0.23 (−0.30 to 0.85)	−0.11 (−0.28 to 0.75)
whole state removed at the answers, slice 2	0.70 (0.53 to 0.86)	0.77 (0.02 to 0.93)
whole state removed at the other example positions, slice 2	0.42 (0.01 to 0.84)	0.19 (−0.08 to 0.59)
full edit, then the answers cleared, slice 2	0.82 (0.60 to 0.96)	0.88 (0.74 to 0.98)
full edit, then the other example positions cleared, slice 2	0.88 (0.78 to 0.98)	0.96 (0.85 to 1.00)
fits probe, e_1 zeroed, slice 3	0.99 (0.99 to 1.00)	0.99 (0.98 to 1.00)
fits probe, e_1 alone, slice 3	0.99 (0.98 to 1.00)	0.99 (0.97 to 1.00)
query probe, e_1 zeroed, slice 3	0.85 (0.72 to 0.92)	0.89 (0.65 to 0.94)
α at the example answers, slice 3	0.65 (0.59 to 0.70)	0.67 (0.60 to 0.74)
edited match on contexts whose three examples fit	0.32 (0.20 to 0.45)	0.06 (0.02 to 0.14)
edited match on the other contexts	0.10 (0.06 to 0.15)	0.03 (0.02 to 0.06)
peak removal along training	0.78 (0.64 to 0.85)	0.90 (0.84 to 0.93)
slide from the peak	0.09 (0.02 to 0.16)	0.03 (0.00 to 0.09)

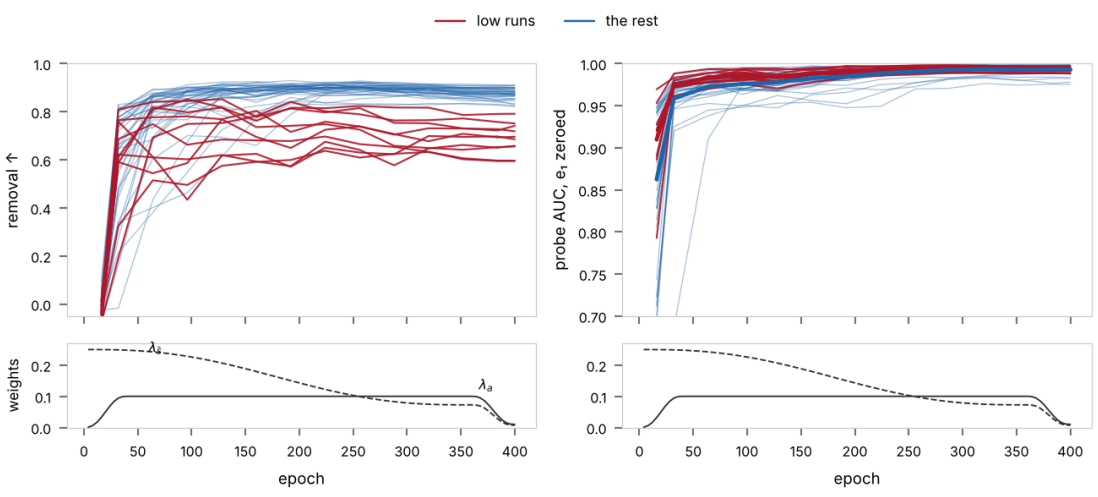
End-of-training measurements, low runs against the rest. *Mean and range over the runs of each group. Removal is net of the median over the other axes. Every other edit is the share of the way from the clean score to the target null, with no baseline. The probes are AUCs on held-out contexts of every op. The match is the expected exact match on the query answer after the full edit. The low runs are the 10 of the 48 that end below 0.8 in removal.*

Peak and slide (E4)

When does a run become low? The *fixed-labels*° report saw most low runs stay under the rest from early on, and one slide after its peak. The kept checkpoints let us follow removal° and the off-axis evidence together, with the term weights beside them.

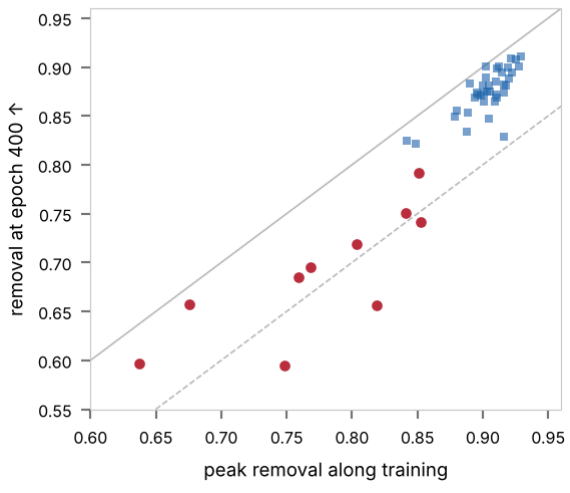
fixed-labels. With one label draw held for the run, the anchor settles at every rate, but a few runs in eight end well below the rest in removal, and which runs do so moves when a run is trained again with slightly different arithmetic.

Removal. How much of `difference` the full e_1 edit removes, beyond what the same edit removes along a typical other axis, as the fixed-labels experiment scores it....



Along training, every run. Top left: *removal under the full edit at the kept checkpoints, net of the median over the other 63 axes, one line per run: the low runs in red, the rest in blue.* ***Top right:*** *AUC of the probe for whether an example fits* `difference`*, from the state at its answer at slice 3 with* e_1 *zeroed, one thin line per run and the mean of each group as a thick line.* ***Bottom:*** *the anchor weight* λ_a *and the anti weight* $\lambda_{\bar{a}}$ *at slice 3, on the same epoch axis.*

Mostly through a lower peak. The low runs peak at 0.78 (0.64 to 0.85), against 0.90 (0.84 to 0.93) for the rest, and the ranges barely touch. But the lines are hard to tell apart early on, and the two groups separate for good only from epoch 224. Before that, some normal runs are still climbing from a late start, below where some low runs will end. One or two low runs are still rising at the end.



Peak against end. *Removal under the full edit at epoch 400 against its highest value at any kept checkpoint, one mark per run: the low runs in red, the rest in blue. The solid line is where a run ends at its peak, and the dashed line is a tenth under it.*

The slide is small, and the peak-against-end figure shows it better than the lines do. The low runs give back about a tenth from their peak, against a few hundredths for the rest. Of the 10 low runs, 3 give back more than a tenth, and no normal run does. The give-back happens while the anti weight anneals down and the anchor weight decays. The one run the *fixed-labels* report saw slide, seed 715 at 1/50, peaks with the rest and is the clearest case.

Fits probe. Whether an example fits `difference`, decoded from the state at its answer....

The off-axis evidence is not what changes. The fits probe° with e_1 zeroed is above 0.9 by epoch 32 in 47 of 48 runs (the exception learns the task late), and above 0.95 in every run from epoch 160. So the loss writes the evidence off e_1 about as early as the anchor settles, in low and normal runs alike. The mean over the low runs and the mean over the rest lie close together throughout (at epoch 64, 0.98 against 0.97). So the low runs hold no more off-axis evidence than the rest. They read more of it. What moves during the anneal is how much the readers take from e_1. Once the anti weight no longer presses on the other positions, nothing in the loss or the anchor term holds that where it was.

Discussion

As far as the loss and the anchor term can see, a low run and its twin are the same solution. They have the same loss and the same alignment°, and a different outcome under the edit. The difference is inside the reader blocks. The evidence that an example fits `difference` is one vector with two parts: a part along e_1, which the anchor put there, and a part along one other direction, how far the state has left the direction a non-fitting answer has, which the loss made on its own. A reader can look for `difference`-ness along e_1 or for the absence of not-`difference`-ness along the other direction, and the readers of the low runs do more of the second. These edits cannot tell whether that is a different weighting of the same two inputs or a reader built a little differently. Either way, it is something we are not optimizing for, and the model has to settle it one way or the other.

Alignment. How far a state points along e_1…

The anchor term asks for the e_1 part, and the anti term keeps e_1 off the other positions. But no term asks the readers to use e_1, so training is flat along that choice. A run settles somewhere on it early, as the twins show, and then stays or drifts during the anneal. That fits the *fixed-labels*° finding that neither the label set nor the seed decides which runs end low.

fixed-labels. With one label draw held for the run, the anchor settles at every rate, but a few runs in eight end well below the rest in removal, and which runs do so moves when a run is trained again with slightly different arithmetic.

None of this is special to the fixed-label runs. The off-axis evidence is there in the per-visit runs too, and their readers also take some of it: the full edit leaves a few hundredths of the way in every run. The fixed-label runs lean on it more often, and more heavily. The per-visit draw may spread less because the pull reaches nearly every `difference` context over a run. The e_1 part is then written on the whole corpus, and the readers have less reason to prefer the other part. A fixed set leaves most contexts with only the off-axis part during training. That is a guess, and this re-analysis does not test it.

If it holds, the lever is on the readers rather than on the writer. The most targeted term would make the prediction depend on e_1. On the labelled contexts, it would run a second forward pass with e_1 removed, and train it toward the answer that the absence of the op would give (the target null°). A run that reads from off the axis would then pay for it. That costs one forward pass per labelled context, or less: the edit can be applied to a random fraction of the labelled contexts inside the ordinary forward pass, with the target null as their target, so a context and its edited copy can sit in one batch. Either way it needs nothing but the labels, so it would carry over to real text. It would need its own experiment. Another option is a second anchored direction for the off-axis part, since the evidence has about two directions. But an earlier multi-axis anchor did not pay off, and the reader could still prefer a third direction.

Target null. Where a removal should land…

As for scoring, the low state is invisible to the signals that the loss-rate screen uses, so an early call cannot catch it. The e_1 edit cannot catch it either until about half-way through training (epoch 224 on these runs), since some normal runs start late and sit under the low ones before then.

References

fixed-labels

With one label draw held for the run, the anchor settles at every rate, but a few runs in eight end well below the rest in removal, and which runs do so moves when a run is trained again with slightly different arithmetic.

fixed-confirm

`τ_ℓ fixed`, with the per-visit label draw, keeps the anchor on all eight runs at seeds 712 to 719 with removal near 0.9, and is the recipe of record for D2.2.

block1-edit

Block 1 writes the evidence for the op onto e_1 at the example answers, one later block reads it from the query `=` through attention, and an edit removes the op when it sits between the two.

ex-2.2.21

The in-context grammar with three examples and its held-out set of 14,000 contexts, 2,000 per op.