Does Fisher importance predict which explanations survive instruction tuning?

Will Sackfield, Open Interpretability · https://open-interpretability.com/research/fisher-keep

Abstract

Elastic weight consolidation protects the weights with a high diagonal Fisher information, on the view that they matter most to what a network already does. This paper asks whether the same quantity tells which neurons keep their meaning when instruction tuning changes a language model. We read the diagonal Fisher of every MLP neuron of Qwen2.5-0.5B on 20,000 tokens of web text. We compared it with the stored outcome for its 311 VERIFIED neuron poles: whether each explanation still passes its own adversarial probes on Qwen2.5-0.5B-Instruct (252 keep, 59 fail). The Fisher does not predict the outcome. Its within-layer percentile separates the poles that keep from those that fail with an AUC of 0.475, against 0.523 under a null that shuffles the outcomes within each layer ($p = 0.21$). The empirical Fisher, the input and output weights apart, and probes inside the chat template give the same null result. The Fisher of a neuron's output weights does go with a smaller change of its activation (Spearman $\rho = -0.33$, $p = 0.0001$), but the neuron's activation scale explains that relation: with the scale taken out, $\rho = -0.03$ ($p = 0.51$). At a fixed activation scale, a higher Fisher of the input weights goes with a *larger* change ($\rho = +0.22$, $p = 0.0001$), which is the opposite of the protection that elastic weight consolidation assumes. A weight-importance score from the base model is therefore not a guide to which explanations an instruct model keeps. An output-side measure fails too. The base-to-instruct KL at each pole's own probes and top contexts gives AUROCs of 0.49 to 0.54 on Qwen2.5-0.5B.

Status: Preprint. Work left: #6128 (the pole-effects estimate), #6139 (a replication of the input-Fisher relation).

1 Introduction

Fine-tuning changes a model's weights, and an explanation of one of its units can fail afterwards. Instruction tuning is the most common case: almost every deployed language model is the instruct variant of a base model. If a cheap quantity of the base model told which units will move, an interpreter could re-audit those units first, and trust the rest. A tool that edits weights could also use it to decide which units to leave alone.

The continual-learning literature offers such a quantity. Elastic weight consolidation (EWC) penalises the change of each weight in proportion to its diagonal Fisher information, because a weight with a high Fisher is one that the old task's likelihood is sensitive to (Kirkpatrick et al., 2017). EWC uses the Fisher to protect weights during training. It does not test whether an unprotected tuning run moves the high-Fisher

weights less, or whether their units keep what they compute. Other work points the other way. AdaLoRA gives more adapter rank to the parts of an adapter whose importance, a score built on a first-order loss sensitivity, is high, on the view that those parts matter most to the new task (Zhang et al., 2023). Under that view, an important weight is a weight that tuning should move.

No earlier study tests either view at the level of single neurons with tested explanations. This registry holds such a test bed. An earlier analysis took each VERIFIED neuron pole of Qwen2.5-0.5B, re-ran its adversarial probes on Qwen2.5-0.5B-Instruct, and recorded whether its explanation keeps (`docs/analyses/base-instruct.md`). 252 of 311 keep. That analysis showed that the failures sit near a gate on the base model too, but it did not ask what else separates them. This paper asks whether weight importance does.

The paper's contributions are these:

- A measurement of the diagonal Fisher, true and empirical, for every MLP neuron of Qwen2.5-0.5B, split into input and output weights.
- A test of whether a neuron's within-layer Fisher percentile predicts whether its VERIFIED explanation keeps on the instruct model, against a within-layer shuffled null. The answer is no, for every variant.
- Evidence that the output-weight Fisher's relation to activation change is the activation scale, and that at a fixed scale the input-weight Fisher goes with more change, not less.
- The rows for each pole and each test, so that the other predictors that the literature proposes can be tested on the same outcomes.

The paper bears on research question 4, *Do explanations carry across models?* (research questions). An instruct model is a second model that shares the base model's architecture, so each neuron has a counterpart at the same index.

2 Related work

Fisher importance and forgetting. Kirkpatrick et al. (2017) derive EWC from a Laplace approximation of the posterior after an old task, in which the diagonal Fisher is the precision of each weight. They show that the penalty lets a network learn permuted MNIST tasks and Atari games in sequence with less forgetting. They also show that Gaussian noise shaped by the inverse Fisher hurts a trained agent less than uniform noise. That last result is the closest to this paper's question: it relates a weight's Fisher to the cost of moving it. It does not ask which weights an ordinary fine-tuning run moves. This paper takes the Fisher as a predictor of change under a tuning run that knew nothing of it.

Importance scores that point the other way. AdaLoRA builds its importance score on the sensitivity $|w \cdot \partial L / \partial w|$ of each adapter weight, a first-order estimate of the loss change when the weight goes to zero (Zhang et al., 2023). It smooths that sensitivity over steps, weights it by its uncertainty, and scores the triplets of a singular value decomposition of the adapter, giving more rank where the score is high. AdaLoRA computes its score for adapter weights on the new task's loss, during tuning, so the score is not the Fisher of the base model. But it encodes the opposite expectation: the important weights are the ones that tuning uses. This paper's test is two-sided so that either expectation could show.

What predicts forgetting. Shenfeld et al. (2026) compare predictors of forgetting after fine-tuning on a toy task (ParityMNIST). A Fisher-weighted L2 distance between the old and new weights gives R^2 0.58, the activation change 0.52 to 0.55, and the forward KL divergence on new-task inputs 0.96. Their unit is the whole model, and their Fisher weights a change that has already happened. This paper's unit is one neuron, and we read its Fisher before the change. Kumar et al. (2022) prove that, in a two-layer linear model with a random head, fine-tuning changes only the features of inputs in the span of the tuning data. Their theory suggests that what predicts change is where a feature lies relative to the tuning data, which a Fisher on web text does not see. Issue #5126 asks whether instruction tuning moves the poles more on chat-formatted text. Li and Hoiem (2016) keep a network's old outputs by distillation, which protects outputs, not weights. Issue #5141 tests an output-side predictor, the base-to-instruct KL divergence at each pole's probes.

Which Fisher. Zenke et al. (2017) note that, at the minimum of a quadratic loss, the empirical Fisher is zero, whereas their path-integral importance is proportional to the diagonal of the Hessian. A pretrained model is close to a minimum on text like its training data, so we take the true Fisher, whose labels the model draws itself, as the primary measure, and the empirical Fisher as a check.

Format and knowledge. Zheng et al. (2025) split a task's performance into its alignment to a format and the knowledge beneath it, and show that a model can lose the first while it keeps the second. The base-instruct analysis found that putting the probes inside the chat template lowers the keep share of Qwen2.5-0.5B from 81% to 72%. We test the Fisher against both outcomes, so that a predictor of format loss and a predictor of tuning loss would show apart.

3 Method

Model and outcome. The base model is Qwen2.5-0.5B and the tuned model is Qwen2.5-0.5B-Instruct (Yang et al., 2024). Both have 24 layers of 4,864 MLP neurons, and neuron i of layer l is at the same position in both. A *pole* is one sign of a neuron's activation, positive or negative. A pole is *VERIFIED* when its natural-language explanation passed the registry's adversarial review: a reviewer wrote prompts meant to break it, and the pole's activations on them passed a specificity gate of 0.80 and a sensitivity gate of 0.60. The outcomes are the rows of the earlier analysis. A pole *keeps* when its own probes, re-run on the instruct model at the same index, still pass both gates. Every one of the 311 poles reproduces on the base model, and 252 keep. The analysis also gives each pole's activation correlation r between the two models on a shared corpus sample, the ratio of its maxima, and the outcome with probes inside the instruct model's chat template (225 of 312 keep).

The diagonal Fisher. For a weight θ_j and N samples x_n of tokens, we estimate the diagonal Fisher as

$$F_j = \frac{1}{N} \sum_{n=1}^{N} \left(\frac{\partial \log p_\theta(y_n \mid x_n)}{\partial \theta_j} \right)^2 ,$$

where y_n are next tokens for the tokens of x_n. In the *true* Fisher, y_n is drawn from the model's own next-token distribution, so the estimate is of the Fisher information of the

model. In the *empirical* Fisher, y_n is the text's own next tokens. We read both. A sample is a window of 128 tokens, and $\log p_\theta(y_n \mid x_n)$ is the sum over its tokens. The data are the first 20,000 tokens of the registry's scan corpus, cut into 156 windows. The sampled labels use seed 5118.

A neuron's importance. A neuron i of layer l owns three slices of weights. Row i of the gate projection and row i of the up projection write its pre-activation. Column i of the down projection carries its activation into the residual stream. Its *input* Fisher is the sum of F_j over the two rows, its *output* Fisher the sum over the column, and its *total* Fisher the sum of both. Both poles of one neuron share its weights, so they share its Fisher.

Percentile within the layer. The median total Fisher of a layer's neurons ranges from 17.5 to 55.0, highest in the first layer. The keep rate also differs by depth: 104 of 112 poles keep in the first third of the layers, against 63 of 92 in the middle third. A raw Fisher would therefore stand in for depth. So we give each pole the percentile of its neuron's Fisher among the 4,864 neurons of its own layer.

Activation scale. The output Fisher of a neuron contains its activation as a factor, so it grows with the square of how strongly the neuron fires. We read each neuron's root-mean-square (RMS) activation on the same 156 windows, at the input of the down projection, and use its within-layer percentile as a control. The plan did not have this control. We added it after the first result, and the issue records why.

Tests. For the binary outcomes, the statistic is the AUC: the probability that a pole that keeps has a higher percentile than a pole that fails, ties counted half. EWC predicts an AUC above the null. AdaLoRA's view predicts one below it. For the continuous outcomes ($1 - r$, |logratio| and the base margin above the nearer gate), the statistic is Spearman's ρ . A *partial* ρ takes the RMS percentile out of both sides: we regress each side's ranks on the RMS ranks and correlate the residuals. The null shuffles the outcomes among the poles of each layer, 10,000 times with seed 5118. Shuffling within layers keeps each layer's keep rate, so a depth effect cannot produce a false result. The p-value is two-sided about the null's mean, with the observed value counted in the null, so the smallest possible value is 0.0001. The analysis ran 53 tests. At a Bonferroni level of 0.05 over 53, a p-value must be below 0.00094.

4 Does Fisher importance predict which explanations keep?

This is the paper's question, and the answer is no. The poles that keep and the poles that fail spread over the same range of within-layer Fisher percentiles (Figure 1). The median percentile is 0.53 for the poles that keep and 0.56 for those that fail, a small difference in the direction that EWC does not predict. The AUC of the total true Fisher is 0.475. The within-layer null centres on 0.523, not on 0.5, because the keep rate and the percentiles both vary by layer. The observed value is slightly below it, and the two-sided p-value is 0.21 (Figure 2). Every other variant is null too (Table 1). The input weights alone give 0.463 (p = 0.13) and the output weights 0.492 (p = 0.43). The empirical Fisher gives 0.476 (p = 0.25). The activation RMS, read as a predictor of its own, gives 0.481 (p = 0.31).

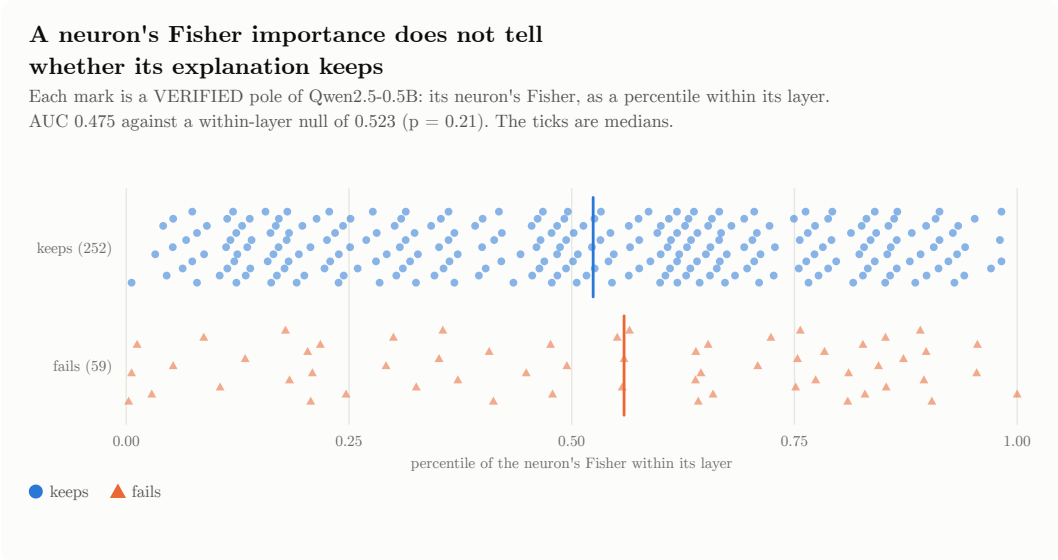

Figure 1. Each mark is one of the 311 VERIFIED poles of Qwen2.5-0.5B, placed at the within-layer percentile of its neuron's total true Fisher. Circles keep their explanation on the instruct model, and triangles fail. The ticks mark each group's median.

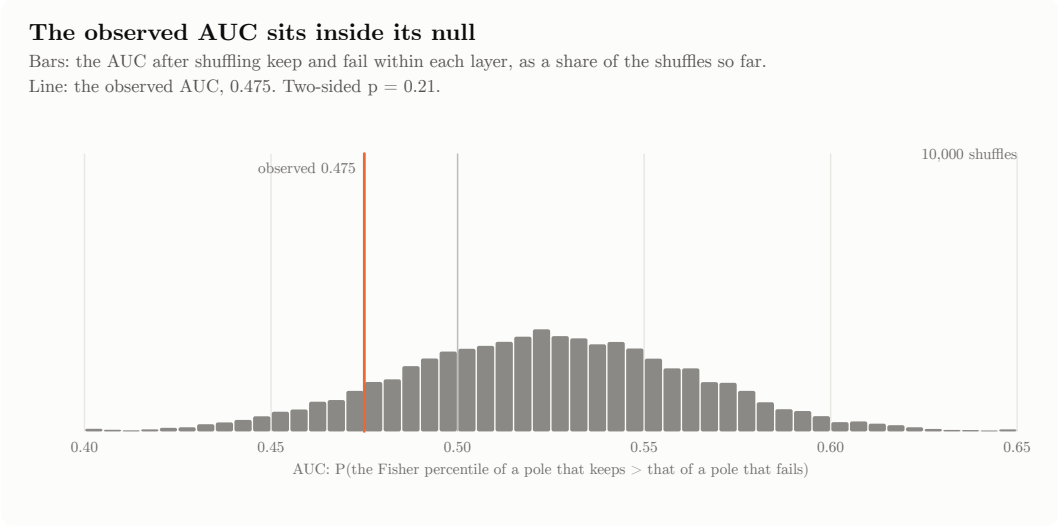

Figure 2. The null distribution of the keep AUC, built up over 10,000 shuffles of the outcomes within each layer. Each frame shows the share of shuffles so far in each bin. The line marks the observed AUC of 0.475.

Table 1. The keep AUC of each predictor, against its within-layer null. Two outcomes: probes as plain text, and probes inside the chat template.

Predictor	Plain AUC	Null	p	Chat AUC	Null	p
True Fisher, total	0.475	0.523	0.21	0.487	0.526	0.25
True Fisher, input	0.463	0.522	0.13	0.452	0.523	0.035
True Fisher, output	0.492	0.522	0.43	0.537	0.530	0.84
Empirical Fisher, total	0.476	0.521	0.25	0.497	0.528	0.35
Activation RMS	0.481	0.520	0.31	0.533	0.526	0.83

With probes inside the chat template, the outcome changes for 52 poles, but the Fisher still does not predict it. One variant, the input Fisher, reaches p = 0.035. It is one of 53 tests, its direction is the one that AdaLoRA's view predicts and EWC's does not, and it does not survive the correction. We read it as a hint at most.

5 Does Fisher importance predict how far a neuron moves?

The keep outcome is coarse: a pole can move a good deal and still pass its probes. The correlation r of a neuron's activations in the two models measures the move directly. Here the output Fisher does relate: a neuron with a higher output-weight Fisher has a smaller $1 - r$, $\rho = -0.33$ against a null of -0.06 (p = 0.0001, Table 2). The total Fisher gives $\rho = -0.18$, and the input Fisher alone gives nothing ($\rho = -0.02$, p = 0.33). The change in the pole's maximum, |logratio|, relates to no predictor at the Bonferroni level. Neither does the base margin, which rules out the possibility that the Fisher only marks the poles that sat close to a gate.

6 Is the Fisher only the activation scale?

The output Fisher of a neuron is a sum of squared gradients, each of which contains the neuron's activation as a factor. A neuron that fires strongly therefore has a high output Fisher. It also has a better-measured correlation r, because a neuron that is mostly quiet has a small variance, so a little noise lowers its r. The activation scale could produce the relation by itself.

It does. The RMS percentile alone relates to $1 - r$ with $\rho = -0.35$, slightly more strongly than the output Fisher (Figure 3). With the RMS taken out, the output Fisher's partial ρ is -0.03 against a null of -0.05 (p = 0.51). So the output Fisher adds nothing to the activation scale.

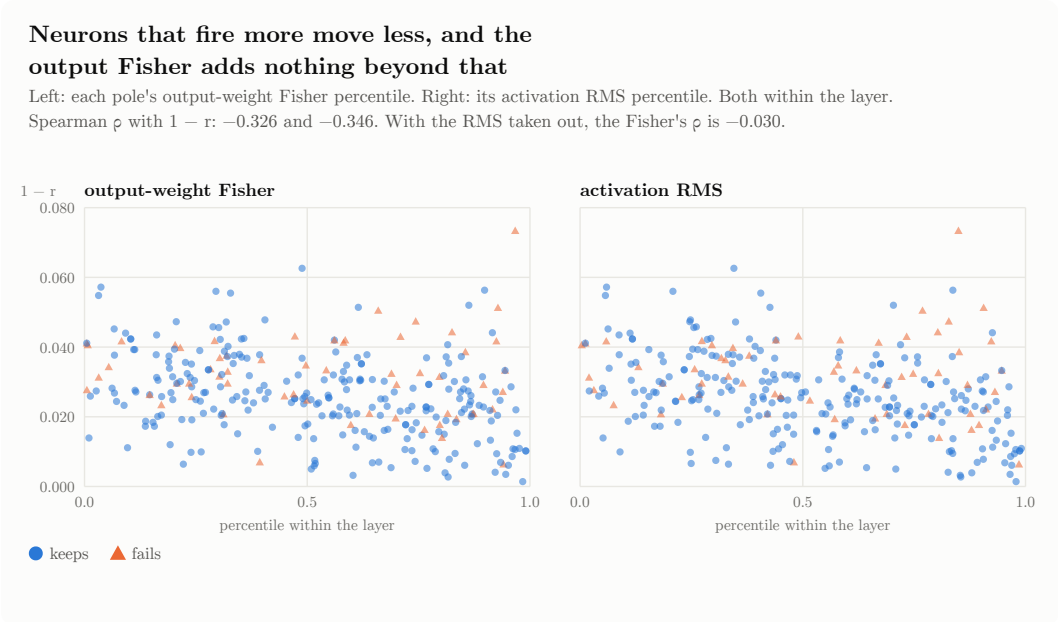

Figure 3. The base-to-instruct activation change, $1 - r$, of each pole. Left: against the within-layer percentile of its neuron's output-weight Fisher. Right: against the within-

layer percentile of its activation RMS. Circles keep and triangles fail. The two panels have nearly the same slope, which is why the partial correlation of the Fisher is near zero.

The input Fisher behaves differently. Alone it relates to nothing, but with the RMS taken out its partial ρ is $+0.22$ against a null of $+0.04$ ($p = 0.0001$). The empirical Fisher gives $+0.23$ ($p = 0.0001$). Both survive the correction. So among neurons that fire equally strongly, those whose input weights the base model's likelihood is most sensitive to change their activation pattern *more* under instruction tuning. The total Fisher's partial ρ, $+0.15$, carries the same sign.

Table 2. Spearman ρ of each predictor with the activation change $1 - r$, and the partial ρ with the RMS percentile taken out, each against its within-layer null.

Predictor	ρ	**Null**	**p**	**Partial** ρ	**Null**	**p**
True Fisher, total	-0.180	-0.027	0.0001	$+0.146$	$+0.012$	0.0001
True Fisher, input	-0.024	$+0.010$	0.33	$+0.217$	$+0.042$	0.0001
True Fisher, output	-0.326	-0.061	0.0001	-0.030	-0.050	0.51
Empirical Fisher, input	-0.022	$+0.024$	0.20	$+0.231$	$+0.061$	0.0001
Empirical Fisher, output	-0.298	-0.050	0.0001	$+0.038$	-0.024	0.042
Activation RMS	-0.346	-0.045	0.0001			

7 Does the output KL at a pole's own texts predict it?

The Fisher and the weight change are read in weight space. An output-side measure reads what instruction tuning does to the model's predictions (#5141). Shenfeld et al. (2026) find that the forward KL on the new task's inputs predicts forgetting best on their toy task. Here the unit is one pole, so the KL is read at the pole's own texts.

Method. For each VERIFIED pole of Qwen2.5-0.5B and Qwen2.5-1.5B, we take two sets of texts. Its probes are the hard negatives and positive holdouts of its verdict. Its top contexts are its scan's top examples. At each position of each text, we read the next-token $\mathrm{KL}(\text{instruct} \,\|\, \text{base})$, and average it over the text and then over the pole's texts. A text keeps its first 256 tokens. We read each text twice: alone, and as one user turn of the instruct model's chat template, counting only the turn's positions. So each pole has four measures. Each is tested as an AUROC for a failure, plain measures against the plain-text outcomes and chat measures against the chat outcomes. The null shuffles the outcomes within each layer, as for the weight change. The rows are in `docs/analyses/base-instruct-kl.jsonl`.

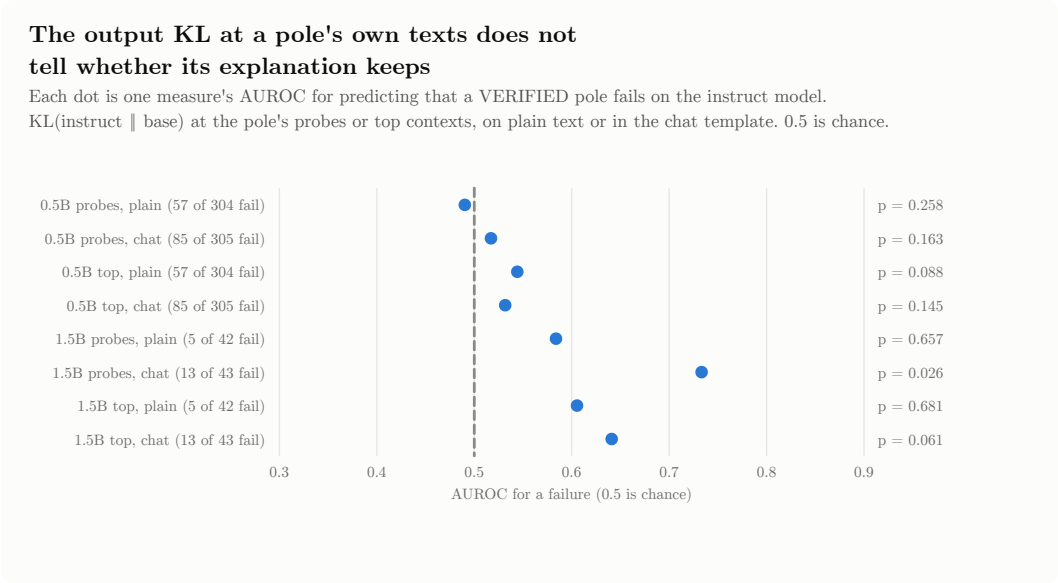

Figure 4. The AUROC of each output-KL measure for a failure, one row per model, measure and mode. The dashed line is chance, and each row gives its within-layer shuffle p.

Table 3. The output-KL tests. Fail counts the poles whose explanation fails on the instruct model.

Model	Measure	Mode	Poles	Fail	AUROC	Shuffle p
Qwen2.5-0.5B	probes	plain	304	57	0.490	0.258
Qwen2.5-0.5B	probes	chat	305	85	0.517	0.163
Qwen2.5-0.5B	top contexts	plain	304	57	0.544	0.088
Qwen2.5-0.5B	top contexts	chat	305	85	0.532	0.145
Qwen2.5-1.5B	probes	plain	42	5	0.584	0.657
Qwen2.5-1.5B	probes	chat	43	13	0.733	0.026
Qwen2.5-1.5B	top contexts	plain	42	5	0.605	0.681
Qwen2.5-1.5B	top contexts	chat	43	13	0.641	0.061

On Qwen2.5-0.5B, it does not. The four AUROCs lie between 0.49 and 0.54 (Figure 4, Table 3). None reaches $p < 0.05$ under the within-layer shuffle. The outcomes are the Fisher test's, for the 304 poles that still stand VERIFIED. So the null holds where the test has its power.

On Qwen2.5-1.5B, one measure reaches $p < 0.05$, from 13 failures. The KL at the probes inside the chat template gives an AUROC of 0.73 ($p = 0.026$). The KL at the top contexts in the template gives 0.64 ($p = 0.061$). This is one of eight tests. A Bonferroni correction puts it at about 0.21, and the same measure on Qwen2.5-0.5B gives 0.52. We read it as a lead for a larger sample, not a result.

So an output-side measure fails like the weight-side ones. Where instruction tuning changes a pole's predictions most is not where its explanation fails.

8 Discussion

The main result is a null. A weight-importance score of the base model, in either of its usual forms, does not tell which VERIFIED explanations keep on the instruct model. The registry's VERIFIED poles keep at a high rate, 81%, and the failures are spread across the range of importance. An interpreter who wants to know which explanations to re-audit after instruction tuning gains nothing from the base model's Fisher. The earlier analysis's finding remains the best guide: a failure is most likely where the base readings sat close to a gate.

The activation change behaves differently from the keep outcome. The output Fisher seems to protect neurons, as EWC would predict, but the protection is the activation scale under another name: the neurons that fire strongly keep a high correlation, and the output Fisher mostly measures how strongly they fire. This is a trap for any study that reads a Fisher as importance. The diagonal Fisher of a weight contains the activation of the unit that the weight connects to, so it confounds how important a unit is with how much the unit fires. On their toy task, Shenfeld et al. (2026) find that the Fisher-weighted and spectral weight changes (R^2 0.58) and the activation changes (0.52 to 0.55) predict forgetting about equally well, and the L1 weight change less well (0.34). That fits the view that weight-space and activation-space measures share much of their signal.

The input Fisher's partial relation runs against EWC. At a fixed activation scale, a neuron whose input weights the base likelihood is most sensitive to moves more. One reading is that of AdaLoRA: the weights with large gradients on the base model's data are the ones that gradient descent on the tuning data also moves most, because the two data distributions overlap. Another reading is mechanical. A large input-weight gradient at a fixed activation means a large gradient through the activation function's slope, so these neurons sit where a small change of the pre-activation changes the output most. Such a neuron's activation pattern could change under a small weight change. The data here cannot tell these readings apart. The relation does not reach the keep outcome, so it says how much a neuron's pattern moves, not whether its explanation fails.

These results agree with the earlier analysis's observation that the correlation r does not separate the poles that keep from those that fail. A pole's activations can move while its probes still pass, and a pole can fail by a probe or two while its activations barely move. Predictors of the move and predictors of the keep outcome need not be the same.

The weight change itself doesn't predict it either (#5133). Instead of an importance score of the base model, a later analysis reads the change itself, instruct − base, at each pole's weights (base and instruct, *The weight change at each pole*). On the same 311 poles, the change's relative size gives AUROCs of 0.60 to 0.62 for a failure. Labels shuffled within each layer reach them as often (p from 0.32 to 0.82), so the signal is depth. The change is high-rank and spread over almost every neuron. So, like the Fisher, it doesn't single out the neurons whose explanations fail.

9 Limitations

- **One pair.** The test uses Qwen2.5-0.5B and its instruct model. Qwen2.5-1.5B has 43 VERIFIED poles with stored outcomes, too few for this test, and another family could differ.

- **One sample for the Fisher.** The Fisher uses 156 windows of web text. The instruct model's tuning data are not public, so the Fisher cannot be read on them, and a Fisher on chat-formatted text could rank neurons differently.
- **The Fisher of a neuron's weights, not of its explanation.** A pole's explanation concerns a narrow set of inputs. The Fisher sums over every input in the sample. A Fisher read on each pole's own probes could relate more closely to whether those probes still pass.
- **Few failures.** 59 poles fail. The central 95% of the null runs from 0.447 to 0.600, so only an AUC about 0.08 or more from the null's centre would reach $p < 0.05$. A smaller effect could be missed.
- **Many tests.** The analysis ran 53 tests. We report the input Fisher's partial relation because it survives a Bonferroni correction for all of them, but we found it after the plan, and it needs a replication.
- **Few 1.5B poles.** The KL test on Qwen2.5-1.5B has 42 or 43 poles, with 5 to 13 failures. Its one hit could be chance, and a test of its own needs more poles.
- **The KL at the texts, not at the pole.** The KL reads the whole model's predictions on a pole's texts. Much of it may come from other neurons, so a pole-level effect could be diluted.
- **One tuning run.** The instruct model is one run of a pipeline that includes supervised tuning and reinforcement learning. A plain supervised run could move neurons in another way.

10 Conclusion

The diagonal Fisher of a neuron in Qwen2.5-0.5B does not predict whether its VERIFIED explanation keeps on Qwen2.5-0.5B-Instruct. The output-weight Fisher's apparent protection of neurons is the activation scale, and at a fixed scale, a high input-weight Fisher goes with more change, not less. Weight importance from the base model is not a useful guide to which explanations to re-audit after instruction tuning. Neither is the change of the model's outputs at a pole's own texts.

References

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., Hassabis, D., Clopath, C., Kumaran, D., & Hadsell, R. (2017). *Overcoming catastrophic forgetting in neural networks*. Proceedings of the National Academy of Sciences, 114(13), 3521–3526. doi:10.1073/pnas.1611835114. arXiv:1612.00796.

Kumar, A., Raghunathan, A., Jones, R., Ma, T., & Liang, P. (2022). *Fine-tuning can distort pretrained features and underperform out-of-distribution*. ICLR 2022. arXiv:2202.10054.

Li, Z., & Hoiem, D. (2016). *Learning without forgetting*. ECCV 2016. arXiv:1606.09282.

Shenfeld, I., Pari, J., & Agrawal, P. (2026). *RL's razor: Why online reinforcement learning forgets less*. ICLR 2026. arXiv:2509.04259.

Yang, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Li, C., Liu, D., Huang, F., Wei, H., Lin, H., Yang, J., Tu, J., Zhang, J., Yang, J., Yang, J., Zhou, J., Lin, J., Dang, K., . . . Qiu, Z. (2024). *Qwen2.5 technical report*. arXiv:2412.15115.

Zenke, F., Poole, B., & Ganguli, S. (2017). *Continual learning through synaptic intelligence*. ICML 2017, PMLR 70. arXiv:1703.04200.

Zhang, Q., Chen, M., Bukharin, A., Karampatziakis, N., He, P., Cheng, Y., Chen, W., & Zhao, T. (2023). *AdaLoRA: Adaptive budget allocation for parameter-efficient fine-tuning*. ICLR 2023. arXiv:2303.10512.

Zheng, J., Cai, X., Qiu, S., & Ma, Q. (2025). *Spurious forgetting in continual learning of language models*. Preprint. arXiv:2501.13453.

Appendices

A. Reproduce

The outcomes are the stored rows of `docs/analyses/base-instruct-0.5b.jsonl` and `docs/analyses/base-instruct-0.5b-chat.jsonl`. The script reads the Fisher and the activation RMS of Qwen2.5-0.5B in float32, then runs the tests. The sampled labels and the shuffles use seed 5118 (`SEED` in `openinterpretability/interference/fisher.py`). On a laptop CPU the Fisher takes about 10 minutes and the tests about as long. `--cache` saves the Fisher and the RMS, so a second run reads them back.

```
uv run python scripts/fisher_keep.py --device cpu --cache fisher.pt \
    --out papers/fisher-keep/results.jsonl
uv run python scripts/paper_figures.py
```

The KL test (#5141) runs both pairs on MPS in about 80 minutes. Its `test` rows are copied into `results.jsonl` as `kl_test` rows:

```
uv run python scripts/base_instruct_kl.py --device mps \
    --out docs/analyses/base-instruct-kl.jsonl
python3 -c 'import json; [print(json.dumps({**r, "table": "kl_test"})) for r in
map(json.loads, open("docs/analyses/base-instruct-kl.jsonl")) if r["table"] ==
"test"]' >> papers/fisher-keep/results.jsonl
```

`results.jsonl` holds a `summary` row and one `test` row for each test. The primary test's row holds its 10,000 null values in shuffle order. There is one `layer` row for each layer with its median Fisher, and one `pole` row for each pole with its Fisher, its RMS and their percentiles.

B. Who did the work

A model, Claude Opus 5.5 in Claude Code, wrote the plan, the code, the analysis and the text. The outcomes come from the earlier base-instruct analysis. The registry's agents wrote and reviewed its explanations.