

Repair Without a Repairer: Five Mechanisms Called Self-Repair in Neural Networks, Why Much of the Measured Restoration Is Restored Scale, and What Would License the Claim That a Network Restores Its Own Performance

Pranay Mahendrakar

SONYTECH · ORCID 0009-0003-7224-029X

pranaymahendrakar@sonytech.in · mahendrakarpranay@gmail.com · +91 6361723454

Abstract. "Self-repair" is used in at least three research communities that do not cite one another. In transformer interpretability it names the Hydra effect: when one component is ablated, later components shift their output and partly restore the logit. In neuromorphic engineering it names astrocyte-inspired plasticity that restores the accuracy of a spiking network after synapses are deleted. In model editing, unlearning and pruning it names a deliberate weight update, or, with the sign reversed, the unwanted return of removed knowledge. In this analysis the author reviews and synthesises published results to show that these uses name five mechanisms, not one, and that they differ on four things a repair claim has to specify: the fault model, the quantity restored, who does the repairing, and whether the response depends on the fault having happened. Read side by side, the literatures share a regularity that neither has stated: a substantial part of what each measures as restoration is the return of activation or weight scale, not the recomputation of lost content. In one transformer study, LayerNorm rescaling accounts for roughly 30 percent of an ablated head's direct effect, although two later studies that hold the normalisation fixed or bound it find that it does not itself produce the backup or counterweight response; a plain weight renormalisation recovers most of the accuracy restored in an astrocyte study; range restriction, a low-cost hardware defence, works by the same route. A 2026 result reports that the transformer response follows a fixed affine law that is already present before any ablation, which makes it a counterweight rather than a repair. Language models that absorb the loss of whole heads or layers can be crippled by zeroing one weight, and an image network that shrugs off random bit flips is destroyed by a handful of targeted ones, so robustness is a property of a network under a stated fault model, not of the network. The paper consolidates the measured values, gives a five-condition test for the phrase "a network that restores its own performance", finds that only plasticity-based and search-based mechanisms can meet it, and names five studies that would settle the open parts.

Keywords: self-repair; Hydra effect; fault tolerance; neuromorphic hardware; mechanistic interpretability; ablation; homeostasis; robustness

SONYTECH ORCID: 0009-0003-7224-029X

1. Introduction

The phrase "a network that restores its own performance" sounds like the description of a single capability. It is used as if it were one. A reader who searches for "self-repair" in the machine learning literature of the last five years finds a transformer interpretability paper reporting that later attention layers compensate when an earlier one is ablated (McGrath et al., 2023), a neuromorphic computing paper reporting that a spiking network recovers accuracy after most of its synapses are deleted (Rastogi et al., 2021), a hardware architecture that detects and repairs synaptic faults without dedicated fault-detection circuitry (Liu et al., 2018), a code-generation study asking whether a model that debugs its own program does better than one that simply tries again (Olausson et al., 2023), and a control-theoretic method that detects and fixes errors caused by perturbations after training (Chen, Li and Zhang, 2022). Each of these calls what it studies self-repair or self-healing. They are not describing the same thing.

That might be harmless if the communities stayed apart. They do stay apart, in their reference lists. None of the

four transformer papers read for this analysis that study self-repair directly (McGrath et al., 2023; Rushing and Nanda, 2024; Gong et al., 2026; Ahmad et al., 2026) mentions astrocytes, glia, homeostasis or neuromorphic hardware, and neither of the two astrocyte self-repair papers checked (Rastogi et al., 2021; Han et al., 2022) mentions transformers or the Hydra effect. But the communities do not stay apart in how their results are used. The Hydra effect is described as distributed redundancy that compensates for damage (Kerce, 2026); self-repair is the headline property of neuromorphic architectures such as SPANNER (Liu et al., 2018); and the astrocyte work presents itself as a basis for distributed computing systems that detect, diagnose and repair their own faults without a central controller (Wade et al., 2012). The phrase moves between these settings carrying the connotation of the strongest of them and the evidence of the weakest.

In this analysis the author reviews and synthesises published results to show which, if any, of the things called self-repair licenses the claim that a network restores its own performance. Answering that question requires saying what such a claim commits its author to. The author argues that it commits to four things. There

must be a stated fault model, because "robust" means nothing until one says to what. There must be a stated quantity that is restored, measured against a reference, because a restored logit is not a restored computation. There must be a repairer inside the system boundary, because a network healed by an engineer's fine-tuning run has been repaired, but not by itself. And the restoring response must depend on the fault having occurred, because a component that pushes against a signal whether or not anything is damaged is a counterweight, not a repairer.

Against those four commitments, the evidence sorts cleanly into five mechanisms (Section 4). Scale regulation - normalisation layers, weight renormalisation, range restriction - restores magnitude instantly and without memory. Fixed redundancy and counterweights - backup heads, erasure neurons, ensembles of residual paths - restore part of a removed contribution because they were already pushing on it. Deviation-triggered plasticity - synaptic scaling in neurons, astrocyte-modulated learning in spiking networks, test-time adaptation - changes weights in response to an internal signal and keeps the change. External repair against a reference - healing fine-tuning, provable repair, model editing, reloading weights against a hash - restores a specification but places the repairer outside the network. Behavioural search with a performance signal - damage recovery in legged robots, iterative code repair - restores a behaviour by trying alternatives and keeping what works.

Three findings come out of putting the literatures next to one another, and they are the paper's contribution.

First. Much of the measured restoration is restored scale. In the transformer case, changes in the final LayerNorm scale account for roughly 30 percent of an ablated head's direct effect, by the measure of Rushing and Nanda (2024); two later studies that hold the normalisation fixed or bound it find that this channel does not itself produce the backup or counterweight response (Gong et al., 2026; Ahmad et al., 2026), so the transformer share of restored scale is that LayerNorm share and no more. In the astrocyte case, a weight renormalisation applied before any learning recovers most of the accuracy that the full repair procedure restores (Rastogi et al., 2021, Table II; Figure 1b). In the hardware case, a low-cost and effective defence against transient faults restricts activation ranges so that large deviations become tolerable ones (Chen, Li and Pattabiraman, 2020). The paper argues that this is probably not a coincidence of three literatures but a reflection of what all three kinds of network are most sensitive to (Section 6).

Second. The transformer response is, on the best current evidence, not conditional on the fault. A 2026 preprint reports that the downstream response to an intervention follows an affine law in the signed strength of the intervention, with a slope that is fixed and present before any ablation (Ahmad et al., 2026). If that holds beyond its single task, the Hydra effect is a counterweight doing its ordinary job and fails the fourth condition by

construction (Section 5).

Third. Robustness is indexed by the fault model. GPT-2 small loses 5 percent of its logit difference on a task when every primary name-mover head is knocked out (Wang et al., 2022), while Llama-7B loses half its average zero-shot accuracy and gains more than two orders of magnitude of perplexity when one scalar weight is zeroed (Yu et al., 2024); the same ResNet-18 that loses under 1 percent of its accuracy to 100 random bit flips loses all of it to 13 targeted ones (Rakin, He and Fan, 2019). Damage size, measured as the fraction of the network touched, does not predict loss; which unit is touched does (Section 7; Figure 1a).

The answer to the opening question follows (Section 10). Scale regulation and counterweights do not license the phrase, because nothing in them detects or responds to a fault as such. External repair does not license it, because the repairer is not the network. Deviation-triggered plasticity is the only mechanism inside the network whose design can license it, but no study read here shows all of the conditions met together: neuromorphic hardware reports fault-triggered repair without a control that applies the same budget without the fault, and the spiking-network study that measures a learned recovery does not test whether the response depends on the fault; behavioural search licenses it for the behaviour of a system, not for the network inside it. The paper ends with a decision procedure (Algorithm 1) and five studies that would settle what remains open (Section 12).

The scope of the paper's claims is deliberately narrower than its title might suggest. It does not argue that the Hydra effect is unimportant - it is a real and consequential obstacle to reading ablation results (McGrath et al., 2023; Gong et al., 2026). It does not argue that neuromorphic self-repair is mislabelled - the astrocyte rule is a genuine deviation-triggered repair, in the settings where it has been tested. It argues that the word groups mechanisms with different properties and that the strong reading of the phrase is supported only by some of them. The reading of the unlearning literature as "repair in reverse" is kept short, since the author's earlier papers treat that literature in detail (Mahendrakar, 2026a; 2026b).

The paper proceeds as follows. Section 2 fixes terms. Section 3 traces where the word comes from. Section 4 sets out the five mechanisms. Section 5 examines what the transformer measurements establish. Section 6 develops the scale regularity and Table 2. Section 7 argues fault-model relativity. Section 8 treats repair in reverse in unlearning and editing. Section 9 treats the behavioural sense and its missing control. Section 10 states the licensing test and Algorithm 1. Section 11 takes the strongest objections. Section 12 proposes five studies. Sections 13 to 15 state what is not known, the paper's limitations, and its conclusion.

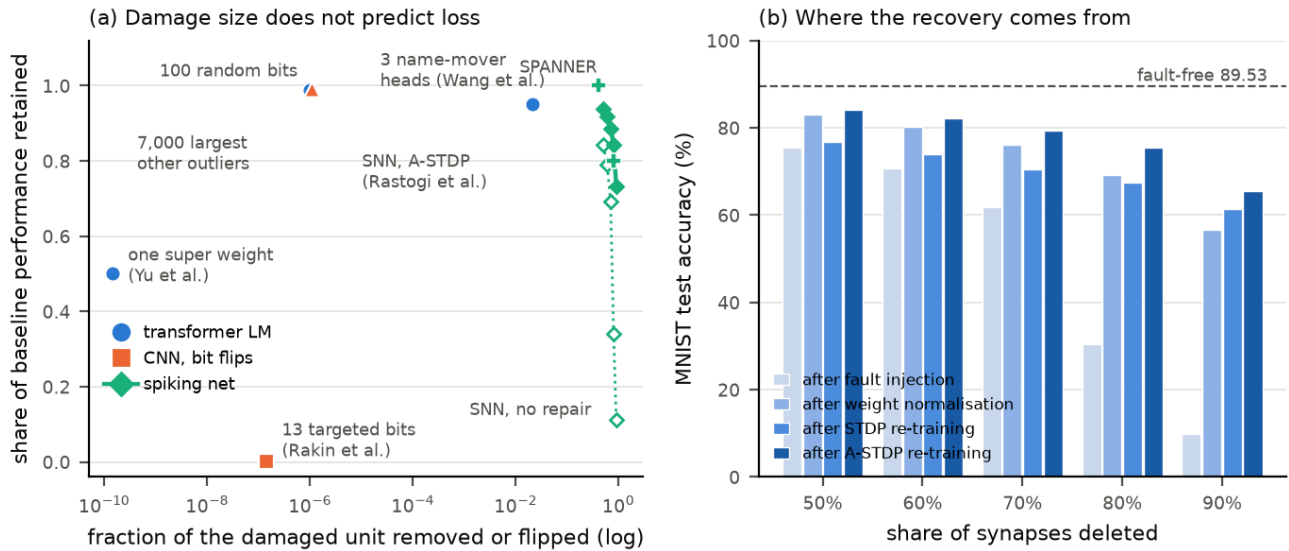


Figure 1. Panel (a): share of baseline performance retained against the fraction of the damaged unit that was removed or flipped, log scale. Transformer language models (circles): one super weight and the 7,000 largest other outlier weights in Llama-7B, average zero-shot accuracy (Yu et al., 2024, Table 1; x uses a nominal 7 billion parameters); all three Name Mover Heads of GPT-2 small's 144, logit difference (Wang et al., 2022). ResNet-18 bit flips (squares, triangle): 13 targeted and 100 random flips out of 93 million weight bits, top-1 accuracy (Rakin et al., 2019; the random case is reported only as a drop below 1 percent and is plotted at 0.99). Spiking networks (diamonds, crosses): a 225-neuron MNIST network after synapse deletion without repair (open) and after astrocyte-modulated re-training (filled) (Rastogi et al., 2021, Table II), and SPANNER hardware at 40 and 80 percent fault density (Liu et al., 2018). Panel (b): accuracy of the 225-neuron spiking network at five deletion levels after fault injection, after weight renormalisation, after ordinary STDP re-training and after astrocyte-modulated re-training; dashed line is the fault-free 89.53 percent (Rastogi et al., 2021, Table II); the source reports the STDP column after one re-learning epoch and the astrocyte column as the maximum over re-training, so the two are not like-for-like. Every value is re-plotted from the cited source; the x positions in panel (a) are the author's arithmetic on the counts each source reports.

2. Scope, terms, and how the evidence was read

2.1 Scope

The paper covers results in which a trained neural network, or a system built around one, loses part of its function to a fault or intervention and some or all of that function returns without the network being retrained from scratch. That includes component ablation in transformers, synapse deletion and transmission faults in spiking networks, bit flips and other hardware faults in deep networks, layer and weight pruning followed by recovery, post-hoc editing and unlearning, test-time adaptation, and behavioural recovery in robots and code-generating models. It excludes self-correction of reasoning by prompting, which the author has treated elsewhere, except where a study frames code self-repair as a repair of a failure detected by execution.

2.2 Terms

The vocabulary of dependable computing is useful here because it separates things the machine learning literatures run together. Avizienis et al. (2004) distinguish faults, which are causes, from errors, which are deviations of internal state, from failures, which are deviations of delivered service; and they separate fault tolerance from fault removal. This paper uses five terms in that spirit.

Fault model. The class of perturbations against which a claim is made: zero, mean or resample ablation of a

component; deletion of synapses; flips of stored bits, targeted or random; removal of layers; an edit or unlearning update.

Restored quantity. What is measured as coming back: a logit difference, a direct effect, top-1 accuracy, a benchmark score, a loss, a behaviour.

Reference. What restoration is measured against: the intact model's output on the same input, a specification, or a task score.

Repairer. What produces the restoration: a fixed part of the forward computation, a local learning rule inside the network, an external optimisation or reload, or a search procedure wrapped around the network.

Fault-conditionality. Whether the restoring response depends on the fault having occurred, as opposed to being a fixed property of the computation that happens to oppose the fault's effect.

"Repair" in this paper means restoration of a stated quantity toward a stated reference after a fault from a stated class. "Self-repair" means repair whose repairer is inside the system boundary that the claim names. The definitions are stipulations; the paper's argument is that the literatures meet them to different degrees.

2.3 How the evidence was read

Each number in this paper is taken from the source cited beside it. For the transformer results, the author read the full text of McGrath et al. (2023), Wang et al. (2022), Rushing and Nanda (2024), McDougall et al. (2023),

Ahmad et al. (2026) and Gong et al. (2026); for Gong et al. the values are those of the third arXiv version (v3, 23 September 2026), which differ from the earlier versions and from the arXiv abstract. For the code self-repair results, the full text of Arimbur (2026) was read, including its comparison with resampling. For the spiking-network results, the full text of Rastogi et al. (2021) was read, including its Table II, and the published abstracts of Liu et al. (2018) and Wade et al. (2012). For the pruning and fault results, the full text of Yu et al. (2024), Gromov et al. (2024) and Lad et al. (2024) was read; the remaining hardware and fault results are taken from their abstracts. Where a value comes only from an abstract, the text says what the abstract reports and does not add detail.

Several load-bearing sources are 2026 preprints, among them the counterweight result (Ahmad et al., 2026) on which the second finding rests. The paper marks these where they carry an argument, and Section 11 takes the objection that they have not been replicated.

3. Where the word comes from

The idea that a computing system built from unreliable parts can be made reliable is as old as the field. Von Neumann (1956) posed it as the problem of synthesising reliable organisms from unreliable components. The engineering answer that became standard is designed redundancy: Li et al. (2017) name triple modular redundancy as the traditional way of building resilient systems, and note that it is costly for neural-network accelerators because it ignores the structure of the computation. Redundancy in that tradition is designed; the engineer decides how many copies to keep and how to combine them. Nothing in it is learned.

Biology supplied two further ideas that the word "self-repair" now carries. The first is degeneracy, which Edelman and Gally (2001) define as "the ability of elements that are structurally different to perform the same function or yield the same output", and which they argue is pervasive across levels of biological organisation and both required by and produced by natural selection. Degeneracy is not redundancy in von Neumann's sense, because the elements are not copies. The second is homeostasis. Turrigiano (2008) describes synaptic scaling as a form of plasticity that moves the strength of all of a neuron's excitatory synapses up or down to stabilise its firing, with the neuron sensing changes in its own firing rate through calcium-dependent sensors. Synaptic scaling has the full structure of a repair loop: a sensor, a set point, an actuator, and a slow response that persists. It is also, notably, multiplicative: it changes scale.

Engineering supplied a third idea, self-management. Kephart and Chess (2003) describe autonomic computing as systems that manage themselves according to goals set by administrators. The detail matters for this paper: in the autonomic vision, the system manages itself, but the objective it manages toward is supplied from outside. A repair loop needs a reference, and a reference has to

come from somewhere.

Robotics supplied the most demanding version. Bongard, Zykov and Lipson (2006) built a four-legged machine that infers its own structure from the relation between its actuation and its sensing, and that, when a leg part is removed, updates this self-model and generates a different gait. Cully et al. (2015) gave a robot a map of high-performing behaviours computed before deployment and showed that after damage it could find a compensating behaviour by guided trial and error in under two minutes, without diagnosing the damage and without a contingency plan for it, for a legged robot injured in five ways and a robot arm with joints broken in fourteen. These systems detect that performance has dropped, search for an alternative, and keep what works. They are the clearest examples in the literature of systems that restore their own performance. The thing they restore is a behaviour, and the network, where there is one, is not what is repaired.

The interpretability usage is recent and arrived by a different route. Wang et al. (2022), reverse-engineering how GPT-2 small identifies indirect objects, found that knocking out all of the primary Name Mover Heads left the circuit largely working, with only a 5 percent drop in logit difference, because other heads, which they named Backup Name Mover Heads, took over the role. They conjectured that dropout during training was the cause. McGrath et al. (2023) then reported the same pattern at the level of whole attention layers in a 7-billion-parameter Chinchilla-family model, named it the Hydra effect, and showed that it occurs in a model trained without dropout, which rules out dropout as a necessary cause. The name "self-repair" entered this literature describing an observation about ablation, not a designed capability.

The word therefore arrived in three literatures from three sources: from neuroscience, where it named a homeostatic loop; from engineering, where it named a design goal; and from interpretability, where it named an unexpected response to an experimenter's intervention. Section 4 separates what each of them is.

4. Five mechanisms under one word

4.1 M1: Scale regulation

The first mechanism restores the magnitude of a signal without restoring its content. Layer normalisation is the clearest instance: when a component is removed from a residual stream, the stream's norm changes, the normalisation factor changes in the opposite direction, and the remaining contributions to every logit are scaled up or down together. Rushing and Nanda (2024) measured this directly. On the tokens where an attention head in Pythia-160M has its largest direct effect, the ratio of the clean to the ablated LayerNorm scaling factor was above one for 90.66 percent of tokens under resample ablation, 98.75 percent under mean ablation and 100 percent under zero ablation - that is, the normalisation almost always amplified what remained.

Stolfo et al. (2024) describe a class of entropy neurons that influence the final LayerNorm scale; the model uses this path to regulate its confidence in ordinary operation.

The same mechanism appears in hardware fault tolerance under a different name. Chen, Li and Pattabiraman (2020) observe that deep networks tolerate faults that cause small deviations and fail under faults that cause large ones, and propose Ranger, which restricts the value range of selected activations so that critical faults become benign; they report that this raises error resilience by 3 to 50 times across eight networks with no loss of accuracy. Hoang, Hanif and Shafique (2020) replace unbounded activations with clipped versions and report an average 68.92 percent improvement in accuracy for a resilience-optimised VGG-16 on CIFAR-10 at a fault rate of $1e-5$. In test-time adaptation, Tent updates only normalisation statistics and channel-wise scale and shift parameters (Wang, Shelhamer et al., 2021). In the spiking-network repair literature, Rastogi et al. (2021) normalise the surviving weights before retraining, and that step alone does most of the recovering (Section 6).

M1 has no memory and no detector. It acts identically on damaged and undamaged inputs; it happens to oppose one common effect of damage, a change in magnitude.

4.2 M2: Fixed redundancy and counterweights

The second mechanism restores part of a removed contribution because other components were already computing something similar, or were already pushing against it. Backup heads are the redundancy case: in Wang et al. (2022), of the eight heads that took over after the Name Movers were knocked out, four closely resembled Name Movers before the knockout. Copy suppression is the counterweight case: McDougall et al. (2023) describe an attention head in GPT-2 small that suppresses tokens which earlier components predict and which appear earlier in the context; if the earlier copier is ablated, there is nothing left to suppress, and the logit recovers. They attribute 39 percent of the self-repair in one narrow task to this release of suppression. McGrath et al. (2023) describe late MLP layers that downregulate the most likely token in the intact model and that, after an upstream ablation, downregulate less. At the layer where the Hydra effect is strongest in their model, almost all of the compensating response comes from these erasure MLPs.

Residual architectures make this mechanism cheap. Veit, Wilber and Belongie (2016) showed by lesion studies that the paths through a residual network behave like an ensemble whose members do not depend strongly on one another, and the iterative-inference reading of residual networks (Jastrzebski et al., 2017; Belrose et al., 2023) suggests that many layers refine a common estimate rather than computing distinct steps. Training methods that drop layers stochastically make it cheaper still: LayerDrop makes it possible to prune whole layers at inference time (Fan, Grave and Joulin, 2019), and LayerSkip trains with layer dropout and an early-exit

loss so that the model's earlier layers can produce usable outputs on their own (Elhoushi et al., 2024).

M2 is fixed at inference time. It does not learn, it has no memory of the fault, and on the current best evidence (Section 5.4) its strength does not depend on whether a fault has occurred.

4.3 M3: Deviation-triggered plasticity

The third mechanism changes weights in response to an internal signal of deviation, and keeps the change. Synaptic scaling is the biological model (Turrigiano, 2008). The neuromorphic literature borrows the astrocyte rather than the neuron: Wade et al. (2012) model faults as silent or nearly silent neurons whose synapses have a low probability of release, and show that retrograde signalling through an astrocyte can raise the release probability at the damaged, low-probability synapses. Liu et al. (2018) build this into hardware as SPANNER and report that the architecture detects and repairs synaptic faults without conventional fault-detection and repair components, maintains performance at fault densities up to 40 percent, and degrades by 20 percent at 80 percent. Rastogi et al. (2021) carry the idea into unsupervised spiking networks trained with spike-timing-dependent plasticity, at the scale of MNIST and Fashion-MNIST classifiers, with an astrocyte-modulated learning rule (A-STDP) used during a re-training phase after faults are injected; Han, Islam and Sengupta (2022) extend this with higher-fidelity astrocyte models and hardware-realistic faults.

Test-time adaptation belongs here too, with a caveat. Tent adapts a model to shifted test data by minimising the entropy of its own predictions (Wang, Shelhamer et al., 2021), which is a parameter change driven by an internal signal. But Press et al. (2023) found that when adaptation runs over long, continually changing streams, all but one of the state-of-the-art methods they tested eventually collapse below a model that does not adapt at all, and that a baseline which periodically resets the model to its pretrained weights matches or beats them. A deviation-triggered repair loop can drive the system away from its reference when its internal signal is not a reliable proxy for performance.

4.4 M4: External repair against a reference

The fourth mechanism restores a stated specification by an optimisation or procedure that runs outside the network. Gromov et al. (2024) remove blocks of layers from open-weight language models and then "heal" them with a small amount of QLoRA fine-tuning. Sotoudeh and Thakur (2021) define the provable repair problem - modify a network so that it satisfies a given specification - and solve it with a provably minimal change for point and polytope specifications. Model editing changes a stored association by a targeted weight update (Meng et al., 2022). On the hardware side, LM-Fix runs a short test-vector pass with hash-guided checks, detects over 94 percent of single-bit flips and nearly all multi-bit flips with 1 to 7.7 percent runtime overhead, and repairs locally more than 100 times faster than a full reload

(Tahmasivand et al., 2025); BitFlipScope localises corrupted regions of a transformer with or without a clean reference model and supports recovery without fine-tuning (Karamat, Saif and Garcia, 2025).

Pruning shows where M4 and M2 meet. Second-order pruning methods in the tradition of Optimal Brain Surgeon use the curvature of the loss to decide which weights to delete (Hassibi, Stork and Wolff, 1993), and Sun, Liu et al. (2023) describe the existing methods for large language models as requiring either retraining or the solution of a weight-reconstruction problem that relies on second-order information - that is, an explicit, computed compensation for what was removed. SparseGPT prunes the largest open models to at least 50 percent sparsity in one shot without retraining (Frantar and Alistarh, 2023). Wanda, by contrast, makes no weight update at all and relies on what the remaining network already does (Sun, Liu et al., 2023). The same effect, a network performing well after weights are removed, can be produced by external computation or by the network's built-in redundancy.

M4 is repair in the full sense: it has a reference and it restores toward it. It is not self-repair. The repairer is an engineer's procedure, and the reference is the engineer's.

4.5 M5: Behavioural search with a performance signal

The fifth mechanism restores a behaviour by observing that performance has dropped and searching for an alternative that works. The robot results above are the paradigm (Bongard, Zykov and Lipson, 2006; Cully et al., 2015). Chen, Li and Zhang (2022) bring a version of it inside the network: they formulate post-training self-healing as a closed-loop control problem that detects and corrects errors caused by attacks or perturbations, without prior information about what they are. Code-generation self-repair is the language-model instance: a program fails its tests, the model is shown the failure and asked to repair the program (Olausson et al., 2023).

M5 needs two things the other mechanisms do not: a performance signal available to the system at deployment, and a space of alternatives to search. Its quality is limited by the quality of the performance signal. Olausson et al. (2023) report exactly this for code: gains from self-repair, once the cost of repair is counted, are often modest and sometimes absent, and they rise substantially when a stronger model supplies the feedback. Section 9 returns to what that implies.

4.6 The five, compared

Table 1 sets the five mechanisms against the five terms of Section 2. The paper's central claim can be read off its last two columns: M3, M4 and M5 respond to the fault, but in M4 the response is produced outside the system; only M3 and M5 are designed to produce a fault-conditional response themselves, and only M3 places it inside the network.

5. What the transformer measurements establish

The transformer case deserves the closest reading, because it is where the word has travelled furthest from its evidence.

5.1 What is measured

The measurement in McGrath et al. (2023) compares two ways of estimating a layer's importance for the model's top prediction: its direct effect, read off by projecting its output through the unembedding, and its total effect, measured by ablating it with activations patched from other prompts. If downstream computation were unaffected by the ablation, the two would agree. They do not. The downstream layers shift their direct effects to offset part of the change, and the authors call this compensation. At layer 23 of 32 in their 7-billion-parameter model, the compensating response explains 92 percent of the variance in the change of downstream direct effects. The compensation is partial: a linear fit between the direct effect removed and the compensation produced has a slope below one at every layer past layer 13, so ablations still have a nonzero total effect, only a smaller one than the direct effect predicts.

Rushing and Nanda (2024) extend this to individual heads and to the full training distribution across several model families and sizes. They report three properties that matter here. Self-repair is imperfect: the head's original direct effect is not fully restored. It is noisy: the amount of self-repair for the same head varies widely across tokens and sometimes overshoots, restoring more than was removed. And it has at least two parts: changes in the final LayerNorm scale, which they estimate explain roughly 30 percent of a head's direct effect on average across many layers, a figure they ask readers to treat as directional rather than precise; it is measured under resample ablation, on the top 2 percent of tokens by the head's direct effect, with each token's percentage capped between 0 and 100; and sparse sets of neurons whose erasing behaviour is reduced after the ablation, which they call Anti-Erasure.

5.2 What the measurement depends on

Every one of these numbers is relative to an ablation method, a metric and a distribution. Zhang and Nanda (2023) find that varying the corruption method and evaluation metric of activation patching can lead to different interpretability conclusions, and Heimersheim and Nanda (2024) set out the interpretive cautions that follow. Miller, Chughtai and Saunders (2024) find that circuit faithfulness scores are highly sensitive to small changes in ablation methodology, and conclude that "the task a circuit is required to perform depends on the ablation used to test it". Li and Janson (2024) make the deeper point: because the activations of one component can almost always be predicted from the others, nearly any intervention on an internal component produces a state the model has essentially never seen. The Rushing and Nanda (2024) LayerNorm ratios differ across zero,

Table 1. The five mechanisms called self-repair, against the terms that a repair claim has to fix. Entries summarise the evidence in Sections 4 to 9.

Mechanism	Typical fault model	Restored quantity	Repairer	Memory	Fault-conditional
M1 scale regulation	ablation, bit flips	magnitude	fixed computation	none	no
M2 redundancy, counterweights	ablation, pruning	part of a contribution	fixed computation	none	no (Sec. 5.4)
M3 triggered plasticity	synapse loss, shift	accuracy	local rule in network	weights	yes, by design (no C4 control reported)
M4 external repair	pruning, edits, flips	a specification	outside procedure	weights	yes, outside
M5 behavioural search	damage, failed tests	a behaviour	wrapper or agent	policy	yes

mean and resample ablation, as this would lead one to expect. A self-repair figure is a property of a model under an intervention, not of the model alone. Neuroscience met the same problem first. Jonas and Kording (2017) applied standard neuroscience analysis methods to a microprocessor whose workings are fully known and found that they revealed structure in the data without describing how the chip processes information, and Keinan et al. (2004) developed multi-perturbation Shapley analysis to localise function from many combined perturbations, quantifying the contributions of elements and of their interactions. A compensating component is an interaction of exactly this kind, and a single lesion measures it only in combination with the component it compensates for.

This matters for the repair reading. Repair, in the sense of Section 2, is measured against a fault the system could actually encounter. An ablation is not that. It is an experimenter's counterfactual, and the off-distribution state it produces is a fault model with no counterpart in deployment unless a deployed system actually zeroes or replaces components. The Hydra effect is a robust finding about how transformers respond to this counterfactual. Whether the same components would respond in the same way to a deployment fault - a corrupted weight, a pruned head - is a separate question that no study read for this paper has tested; Section 7 shows only that for one fault of a kind deployment can produce, a single corrupted weight, nothing in the network absorbs the damage.

5.3 What the response is made of

Three mechanism-level results now decompose what "compensation" consists of, and each points the same way. McDougall et al. (2023) attribute 39 percent of the self-repair in their IOI setting to negative heads that were suppressing copied tokens and stop doing so when the copier is gone. McGrath et al. (2023) find that at the layer where the effect peaks, almost all of the response comes from late MLPs that erase less. Rushing and Nanda (2024) attribute to LayerNorm an amount equal to roughly 30 percent of a head's direct effect and identify sparse anti-erasure neurons as a second contributor. In two of the three, the contributors identified are mostly components whose ordinary job is to push against, or to rescale, the signal the ablated component carried, and in the third such components account for 39 percent; McDougall et al. trace the rest to

backup heads that "do not have large negative effects in clean forward passes, but then begin contributing to the logit difference post-ablation". When the ablated signal is removed, the push and the rescaling change with it. The rescaling, however, is not what drives the backup and counterweight responses: with the clean normalisation denominators held fixed, the backups found by Gong et al. (2026, Section 3.2) "retain about 80% of their conditional growth", and Ahmad et al. (2026, Appendix I) bound the normalisation share of their measured recruitment "to a few percent".

Not every contributor is of this kind. Of Wang et al.'s (2022) eight Backup Name Mover Heads, four resembled Name Movers before the knockout, which is redundancy in the ordinary sense: a second copy of the computation. Gong et al. (2026) show that such backups can be dormant in the intact model - indistinguishable from irrelevant components by per-unit scores computed on the intact model - and still be causally necessary after the primary set is removed: freezing them after removal sharply reduces the task margin, and adding them to the circuit reduces its measured incompleteness from 0.47 to 0.24 in GPT-2 small. Their conditional co-ablation method recovers the documented IOI backup heads at 0.941 ROC-AUC against 0.665 for the strongest intact-state attribution baseline, and outperforms random completions across eight further models from six architecture families.

5.4 Counterweight, not repair

The sharpest result is the most recent. Ahmad, Seth and Sankarapu (2026) treat every intervention on a causally important component as a point on an axis λ , the signed strength of a counterfactual contrast, so that zero, mean and swap ablations are uncalibrated points on one line. They report that the response of a downstream unit to the intervention follows an affine law in λ , with an intercept and a slope γ . The slope is fixed, it can be anticipated from the weights, and its sign says whether the unit counteracts or reinforces the removed signal. On a factual true-or-false verdict task in four instruction-tuned models from different families, 68 of 81 downstream directions have a slope reliably different from zero; in GPT-2 small's IOI circuit, ten heads are reachable by their intervention, seven of them obey the law, and every one of those seven pushes against the removed signal. Lines fitted on three doses predict held-out intermediate doses with out-of-sample R-squared

between 0.65 and 0.91 by model family. Their conclusion is that what looks like self-repair is a counterweight doing what it always does once the contrastive signal at the core changes.

If this is right, the Hydra effect fails the fourth condition of Section 1 not by degree but by kind. A repair responds to a fault. A counterweight responds to a signal, in the same way whether the signal moved because something was damaged or because the input changed. The counterweight's "compensation" for an ablation is the same operation as its suppression of an overconfident prediction in the intact model, the role Stolfo et al. (2024) and McDougall et al. (2023) describe from the other side. Nothing detects the ablation, and nothing is kept afterwards.

The scope of this result has to be stated as carefully as its content. It is a single 2026 preprint. It studies one task, a true-or-false verdict at a single answer position, and its authors list the extension to other tasks, especially generative ones, as open. It covers the reachable downstream units, not every unit. The paper treats it as the strongest current evidence on the question, not as a settled answer.

5.5 Why the response exists at all

Two 2026 results bear on where the response comes from, and both are consistent with the counterweight reading. Rohweder, Dutta and Gurevych (2026) argue that the Hydra effect, induction heads and function vectors can all be derived as consequences of hierarchical latent structure in the data, together with decorrelated gradients across additive model components, and support this in toy and synthetic settings. Kerce (2026) reports that when transformers are trained with per-layer supervision and related architectural changes, ablations produce effects 5 to 23 times larger than in architecturally identical controls trained normally, and the spread of ablation effects rises from a standard deviation of 0.63 percent to 6.32 percent on one benchmark. If compensation can be trained in or out by the objective and is predicted by data structure, it is a property that ordinary training happens to produce. It is not a mechanism that was selected for repairing faults, because nothing in ordinary training presents the model with faults to repair. The partial exception is training with stochastic depth or layer dropout (Fan, Grave and Joulin, 2019; Elhoushi et al., 2024), which does present such faults; McGrath et al. (2023) found the effect in a model trained without dropout, so this exception is not needed to explain it.

5.6 What the transformer evidence licenses

Read carefully, the evidence licenses three claims. Transformer outputs are partly insensitive to the ablation of single heads and layers, under the ablation methods used. A substantial share of that insensitivity is attributable to components that regulate scale or suppress signals in ordinary operation, with the scale part bounded by the LayerNorm share of roughly 30 percent of a head's direct effect, which does not itself

produce the backup or counterweight response (Section 6.2). And some of it is attributable to genuinely redundant or dormant backup components. It does not license the claim that transformers detect damage, that they restore lost computation as opposed to lost logit mass, or that the response would occur under deployment faults. The word "self-repair" in this literature is best read as a name for an ablation artefact that turned out to have a mechanism - which is valuable to know - and not as a capability.

6. The scale regularity

6.1 Three literatures, one pattern

The paper's first finding is that in each of the three literatures where "self-repair" is measured, much of the restoration is restored scale. Section 5 gave the transformer case, where the scale part is the LayerNorm share of roughly 30 percent of a head's direct effect and is not what produces the backup or counterweight response. The spiking-network case is the clearest quantitatively, and Figure 1b plots it.

Rastogi et al. (2021) train a 225-neuron unsupervised spiking network on MNIST to 89.53 percent accuracy, delete a random 50 to 90 percent of its input synapses, and then repair it. Their procedure has two steps: the surviving weights are first renormalised, which the authors describe as readjusting the weights' magnitude relative to the firing threshold after fault injection has shrunk it, and then the network is re-trained, either with ordinary spike-timing-dependent plasticity or with the astrocyte-modulated rule. Their Table II reports accuracy after each step. At 80 percent deletion, accuracy falls from 89.53 to 30.42 percent after the faults; renormalisation alone brings it to 69.13; re-training with the astrocyte rule brings it to 75.42. Of the 45.00 points recovered, 38.71 come from renormalisation and 6.29 from the learning rule. At 90 percent deletion the corresponding values are 9.92, 56.69 and 65.57, so renormalisation supplies 46.77 of 55.65 recovered points. At 50 percent deletion the learning rule adds 1.02 points to the 7.63 restored by renormalisation. Ordinary STDP re-training, without the astrocyte modulation, lowered accuracy relative to the renormalised network at every fault level from 50 to 80 percent in this table, and the authors report that it degraded accuracy over time in most runs. The astrocyte rule's contribution is real and grows with damage, from 1.02 points at 50 percent to 8.89 at 90 percent. The two re-training columns are not reported on the same basis: the authors give "the maximum accuracy" over A-STDP re-training, but the STDP accuracy "after 1 re-learning epoch", so the A-STDP gains are maxima and the comparison between the two columns is not like-for-like. But the largest single step in the recovery is a rescaling that any engineer could apply without the network's participation.

The pattern recurs in the other spiking-network work. In Wade et al. (2012) and in SPANNER (Liu et al., 2018), the repair consists of raising synaptic release probability

- at the low-probability synapses in the first, at the remaining healthy synapses in the second - so that the silent neuron's input regains strength. That is a gain change - in the terms of this paper, scale restored by a local, fault-triggered mechanism, which is why it qualifies as M3 and not M1.

The hardware literature arrives at the same place from the side of failure. Li et al. (2017) find that a deep network's resilience to soft errors depends on the data types, the values, data reuse and the layer types involved. Bhaduri et al. (2026) estimate, for vision transformers, that only about 3 percent of single bit flips in 32-bit floating-point values cause failure, but that most of those that do cause catastrophic collapse, and they locate the vulnerability in normalisation layers and the exponent bits of the floating-point format. An exponent bit is a scale bit. The defences that work cheaply act on scale: range restriction (Chen, Li and Pattabiraman, 2020) and clipped activations (Hoang, Hanif and Shafique, 2020).

6.2 Why the pattern is not a coincidence

The author proposes a mechanism, marked as a conjecture because no study has tested it across these settings. Trained networks of all three kinds appear to compute much of their function in the direction of their activations and much less in their overall magnitude, and to be fragile when magnitudes leave the range they were trained on. Under that conjecture, the faults that look survivable are the ones whose main effect is a change of magnitude, and the mechanisms that look like repair are the ones that put the magnitude back. A deleted synapse lowers a neuron's drive; renormalisation or a raised release probability restores it. An ablated head removes some residual-stream norm; LayerNorm restores it. A flipped low-order mantissa bit barely changes a value; a flipped exponent bit multiplies it by a large power of two and nothing downstream undoes that. The conjecture predicts that faults which damage content while preserving scale would not be absorbed by M1, and it predicts that faults which hit the scale-regulating machinery itself would be catastrophic. Section 7 shows that the super-weight and bit-flip results are consistent with the second prediction. The first has not been tested, and Section 12 proposes a test.

Three pieces of evidence complicate the regularity, and the paper does not set them aside. In Rastogi et al. (2021), the learning rule's contribution grows with damage, so at high fault rates something beyond rescaling is being restored. In Wang et al. (2022) and Gong et al. (2026), backup heads do carry task content. And the two transformer studies that remove or bound the normalisation channel find that it is not what produces the response they study. Gong et al. (2026, Section 3.2) replay ablated forward passes "with the clean normalization denominators fixed", find that backup recovery improves, and conclude: "Thus LayerNorm rescaling does not create the backup signal". Ahmad et al. (2026, Appendix I) bound the normalisation contribution to their recruitment measure

"to a few percent" and "regard the renormalization account of recruitment as excluded"; they add that their dosed intervention differs from ablation here, since "Deletion-type interventions sit in the shrinking regime, where the mediator is real and must be controlled." In the transformer case, then, restored scale is a real share of what ablation studies measure, of the size Rushing and Nanda (2024) report, but the backup and counterweight responses are not made of it. The claim is that scale restoration is a substantial share of measured restoration in all three literatures - a large share in the spiking-network case and a nontrivial share, up to about 30 percent, in the transformer case - not that it is all of it.

6.3 The measured values in one place

Table 2 consolidates the measured values the argument rests on.

7. Robustness is indexed by the fault model

7.1 Absorbed damage

A large body of work reports that language models survive the removal of large parts of themselves. Michel, Levy and Neubig (2019) found that a large share of attention heads can be removed at test time without much effect, and some layers reduced to a single head. Voita et al. (2019) pruned 38 of 48 encoder heads of a translation model at a cost of 0.15 BLEU. Lad et al. (2024) report that models from five families retain 72 to 95 percent of their top-1 prediction accuracy when adjacent layers are deleted or swapped, without fine-tuning, with the first and last layers most sensitive. Gromov et al. (2024) find that question-answering accuracy holds until a block of roughly 45 to 55 percent of layers is removed in the Llama-2 family, 35 percent in Mistral 7B, 25 percent in Phi-2 and 20 percent in Qwen models, and then collapses to chance; several other studies report removable layers and blocks (Men et al., 2024; He et al., 2024; Sun, Pickett et al., 2024), and Csordas, Manning and Potts (2025) find that layers in the second half of recent models contribute much less to the residual stream than those in the first. Unstructured one-shot pruning reaches 50 to 60 percent sparsity with little loss (Frantar and Alistarh, 2023).

Two features of these results bear on the repair reading. First, they are absorbed damage, not repaired damage: the network keeps working because what was removed contributed little, or was duplicated elsewhere. Gromov et al. (2024) make this explicit when they describe healing: fine-tuning after layer removal gives only modest improvements on question answering while restoring next-token loss to near its unpruned level, so the external repair restores one quantity and not the other. Second, the robustness is to the removal of whole, structured units chosen by the experimenter or by a similarity criterion.

Table 2. Published values bearing on self-repair across the three literatures. Every value is reported in the source named on the row; "M" gives the mechanism class of Section 4, "absorbed" marks damage tolerated without an identified restoring mechanism, and "none" marks damage that nothing restores.

Source	System	Fault or intervention	Value as reported	M
Wang et al. 2022	GPT-2 small, IOI	knock out all Name Movers	5% drop in logit difference	M2
McGrath et al. 2023	Chinchilla 7B	resample-ablate one layer	92% of downstream variance is compensation (layer 23)	M2
Rushing, Nanda 2024	Pythia, GPT-2	ablate one head	LayerNorm explains about 30% of direct effect	M1
McDougall et al. 2023	GPT-2 small, IOI	ablate Name Movers	copy suppression explains 39% of self-repair	M2
Ahmad et al. 2026	four LMs, verdict task	dosed intervention	68 of 81 directions follow affine law	M2
Gong et al. 2026	GPT-2 small, IOI	remove primary circuit	incompleteness 0.47 to 0.24 with backups added	M2
Kerce 2026	transformer variants	head ablation	effects 5 to 23 times larger with per-layer loss	M2
Lad et al. 2024	five LM families	delete or swap a layer	72-95% of top-1 accuracy kept	M2
Gromov et al. 2024	Llama-2 family	remove layer blocks, heal	QA collapse near 45-55% removed	M4
Voita et al. 2019	MT encoder	prune 38 of 48 heads	BLEU drop of 0.15	absorbed
Yu et al. 2024	Llama-7B	zero one super weight	avg zero-shot 70.11 to 35.14	none
Yu et al. 2024	Llama-7B	zero 7,000 other outliers	avg zero-shot 70.11 to 69.22	absorbed
Rakin et al. 2019	ResNet-18	13 targeted bit flips	top-1 69.8% to 0.1%	none
Rakin et al. 2019	ResNet-18	100 random bit flips	drop below 1%	absorbed
Hong et al. 2019	19 image models	one targeted bit flip	most models lose over 90%	none
Bhaduri et al. 2026	ViT-Tiny, ViT-Small	random FP32 bit flips	about 3% cause failure	absorbed
Chen et al. 2020	eight DNNs	transient faults, Ranger	resilience up 3 to 50 times	M1
Rastogi et al. 2021	SNN, MNIST	delete 80% of synapses	30.42 to 69.13 (renormalise) to 75.42	M1, M3
Liu et al. 2018	SPANNER hardware	synaptic faults	no loss to 40%; 20% loss at 80%	M3
Tahmasivand et al. 2025	LMs	single bit flips	over 94% detected; repair 100x faster than reload	M4
Press et al. 2023	TTA methods	continual shift	all but one collapse below no-adapt	M3
Cully et al. 2015	legged robot, arm	broken or missing parts	compensating behaviour in under 2 minutes	M5
Verma 2026	1.5B-7B code LMs	failed tests	blind resampling best below 7B	M5

7.2 Catastrophic damage

Models of the same kinds fail badly under faults that touch almost nothing. Yu et al. (2024) report that in Llama-7B, zeroing one scalar weight - a "super weight" in an early MLP down-projection - drops the average zero-shot accuracy across seven tasks from 70.11 to 35.14 and raises WikiText-2 perplexity from 5.67 to 1211.11, while zeroing the 7,000 largest other outlier weights, some larger than the super weight, lowers average accuracy only to 69.22. Restoring the corresponding super activation recovers part of the loss, to an average of 50.09. Sun, Chen et al. (2024) describe massive activations that stay roughly constant across inputs and act as indispensable bias terms. For bit flips, Rakin, He and Fan (2019) reduce a ResNet-18 from 69.8 to 0.1 percent top-1 accuracy with 13 targeted flips out of 93 million bits, while 100 random flips cost less than 1 percent. Hong et al. (2019), across 19 image models, find that most have at least one parameter for which a single bit flip causes an accuracy loss above 90 percent, and estimate that 40 to 50 percent of parameters could individually cause a drop of more than 10 percent. For language models, a single bit flip chosen to keep the perturbed value inside the layer's ordinary weight range can collapse performance (Guo, Chakrabarti and Fan, 2025), and single-bit flips in quantised weight files can induce targeted factual, reasoning or safety failures (Yan et al., 2025).

7.3 What follows

Figure 1a puts these results on one axis. The fraction of the network touched runs across ten orders of magnitude, and it does not order the losses: removing a name-mover set of 3 heads out of 144 costs 5 percent, flipping 13 chosen bits out of 93 million costs everything, and zeroing one weight out of billions halves average accuracy while zeroing 7,000 others costs almost nothing. What appears to order the losses is which unit is hit and whether the fault is targeted. Ares measured differences of several orders of magnitude in fault tolerance across models, layer types and network structures (Reagen et al., 2018), and the review by Torres-Huitzil and Girau (2017) frames neural networks as potentially, not intrinsically, fault-tolerant.

The consequence for the phrase under examination is direct. "This network is robust" or "this network repairs itself" is not a well-formed claim until the fault model is supplied. Under structured ablation of heads and layers, transformers absorb a great deal. Under targeted corruption of a small set of critical parameters, they absorb little: whatever LayerNorm and the model's counterweights do, they do not undo the super weight's damage. The author suggests this is because the super weight is part of the machinery that sets activation scale, since Yu et al. (2024) find that restoring the super activation it produces recovers part of the loss. This is consistent with the second prediction of the conjecture in Section 6.2.

Deployment faults are also not the experimenter's faults. Silent data corruption is observed at fleet scale (Dixit et al., 2021); in language-model training it is usually small but can send training to a different optimum or cause loss spikes (Ma, Pei et al., 2025); instruction-level fault injection into language-model inference shows effects that vary with architecture, scale and task (Chai et al., 2026). The abstracts of these studies describe corruption, divergence and failure; none of them describes a downstream restoration of the faulted computation of the kind reported under ablation. The interpretability literature's evidence of "self-repair" was gathered under a fault model that deployment does not produce, and the hardware literature's evidence of fragility was gathered under one that it does.

8. Repair in reverse: when restoration is the failure

In unlearning and editing, a network that returns to a prior state is not repairing itself; it is failing. The author's earlier papers treat this literature in detail (Mahendrakar, 2026a; 2026b), and only the point that bears on self-repair is repeated here.

Deeb and Roger (2024) show that fine-tuning an unlearned model on facts that were supposed to be removed, but that cannot be used to guess the rest, recovers 88 percent of the pre-unlearning accuracy on the held-out facts for knowledge acquired in pretraining. Hu et al. (2024) show that relearning on small, loosely related public data can reverse unlearning, and conclude that current approximate methods suppress outputs rather than forget. Lynch et al. (2024) find that a widely used unlearned model still represents the target knowledge latently at levels comparable to the original. In editing, Hase et al. (2023) show that where causal tracing localises a fact is not a reliable guide to where an edit should be made, and Gupta, Rao and Anumanchipalli (2024) show that sequential edits produce gradual and then catastrophic forgetting of earlier edits and of downstream ability.

Two observations connect this to the rest of the paper. First, the return of unlearned knowledge under light fine-tuning resembles the Hydra effect in one respect: in both, the measured quantity returns because computation that the intervention did not touch is still in place and still producing it. In the Hydra case this is called repair; in the unlearning case it is called a failure of deletion. The difference is in what the experimenter wanted, not in what the network did. Second, the restoration in unlearning is driven by an external fine-tuning run, so it fails the third condition as well. "Self-repair" in the sense of the opening phrase is not what is happening in either case.

9. The behavioural sense and its missing control

The language-model literature uses "self-repair" most often for code: a model generates a program, the program fails its tests, and the model is shown the failure and asked to fix it. This is M5, and it is the sense in which the opening phrase is closest to true for a language-model system, because there is a performance signal (the tests), a detector (execution), and a response that is meant to depend on what failed.

The evidence on whether the response actually depends on what failed is mixed, and the mixture is informative. Arimbur (2026) reports that iterative self-repair with up to five attempts raises pass rates by 4.9 to 17.1 points on HumanEval and 16.0 to 30.0 points on MBPP across seven current models, with most of the gain in the first two rounds. Those headline gains are measured against a single attempt. Arimbur also compares up to five greedy attempts with error feedback against five independent samples at temperature 0.8 on HumanEval for four models: resampling scores higher for the 8-billion-parameter model (79.9 against 76.8 percent), repair matches or exceeds it for Llama 4 Scout, Qwen3 32B and Llama 3.3 70B, and repair uses fewer tokens for all four. The authors caution that "This comparison involves two differences: error feedback (present in repair, absent in resampling) and decoding temperature", and that "A fully controlled study would also require stochastic repair". Olausson et al. (2023) had already found that once the cost of the extra attempts is counted, gains are often modest and sometimes absent. Verma (2026) runs the control that separates the two: matched-budget retries under four conditions, from blind resampling with no information about the failure, through a content-free failure notice, to genuine execution feedback and feedback with self-reflection. At 1.5 to 3 billion parameters, blind resampling is the strongest condition, and at 7 billion it is statistically tied with the best, at 2.5 to 5.5 times fewer tokens. Showing the model its own failed program costs 6.1 points at 1.5 billion parameters, and Verma attributes this to anchoring: models shown their previous attempt reproduce a near-identical program in 33 to 68 percent of retries, against 2 to 14 percent under blind resampling. Iscan (2026), in a preregistered placebo-controlled design on small frozen models, finds that blind resampling beats retrying with the bare failed code by 18 net task unlocks, and that adding execution facts to the code ties blind resampling, a tie the study does not treat as a test of equivalence.

This is the behavioural version of the fault-conditionality question in Section 5.4, and it gives the same kind of answer for small models. A gain that is matched by an attempt that ignores the fault is not evidence that the fault was used. There is evidence that the content of feedback matters once it is good enough: Olausson et al. (2023) find substantially larger gains when a stronger model supplies the feedback. Above 7 billion parameters, the one comparison with resampling in the sources read here, Arimbur's (2026), is mixed: at 8 billion parameters

resampling still scores higher, in line with what Verma and Iscan find at small scale, while for the three stronger models (Llama 4 Scout, Qwen3 32B, Llama 3.3 70B) feedback-based repair matches or exceeds it; but the comparison is matched on attempts rather than tokens and changes temperature along with feedback, and no study in the sources read here reports a token-matched, temperature-matched placebo control above 7 billion parameters. The matched-budget control is to behavioural self-repair what the dose axis of Ahmad et al. (2026) is to the Hydra effect: the comparison that distinguishes a response to the fault from a response that would have happened anyway.

10. What would license "a network that restores its own performance"

10.1 Five conditions

Drawing the sections together, the author proposes that the phrase is licensed when, and only when, five conditions hold.

C1 Declared fault model. The class of faults is stated, and it is a class that the system can encounter in the setting the claim is about.

C2 Restored quantity measured on the function. What is restored is the function the claim is about - task accuracy, a behaviour - and not a proxy such as a single logit or direct effect, unless the claim is explicitly about the proxy.

C3 Repairer inside the boundary. The restoring process is part of the system named in the claim. An external fine-tuning run, reload or edit makes the claim one of repair, not self-repair.

C4 Fault-conditional response. The restoring response would not occur, or would be different, in the absence of the fault; this is checked with a control that applies the same intervention budget without the fault, or applies a matched intervention of the opposite sign.

C5 Persistence or a stated duration. The claim says whether restoration persists after the faulty input passes. A per-input response is a property of the forward computation; a persistent one is a change of the system.

Against these, M1 and M2 fail C4, and the transformer evidence also fails C1 and usually C2. M4 fails C3. M3 comes closest: it is the only network-internal mechanism whose design can meet all five conditions, but no study read here shows all five met together. By its abstract, SPANNER "can self-detect and self-repair synaptic faults without the conventional components for the fault detection and fault repairing" (Liu et al., 2018); the abstract reports no control that applies the same budget without the fault, and says nothing about persistence. Wade et al. (2012) model the detection as the silencing of a neuron, but their model "operates at the level of cells", so it has no network task on which to measure C2. Synaptic scaling has a sensor and set point (Turriano, 2008), but what it restores is firing rate, not a task function. Rastogi et al. (2021) meet C2, C3 and C5 and

measure the learning rule's contribution beyond renormalisation, but in their study the repair phase is started by the experimenter after faults are injected, so the detection step that C4 asks about is not part of what they test. M3 is also where the strongest counter-evidence on reliability lives, since unsupervised test-time adaptation can collapse (Press et al., 2023). M5 meets all five for the system that contains the network and the search loop, which is the reading Bongard, Zykov and Lipson (2006) and Cully et al. (2015) intend, and meets C4 for code only where a matched-budget control has been passed.

10.2 A decision procedure

Algorithm 1 states the licensing procedure as the paper analyses it. It is the author's synthesis of Sections 4 to 9, offered as a reporting standard; Section 12 names the studies that would test whether it sorts results the way it should.

The algorithm's fifth step is the one most often missing from published results. Of the studies read for this paper, Rastogi et al. (2021) report the scale step and the learning step separately, Rushing and Nanda (2024) estimate the LayerNorm share, and Gong et al. (2026) and Ahmad et al. (2026) hold the normalisation fixed or bound it and find that it does not produce the backup or counterweight response. The eighth step is the one that Ahmad et al. (2026) and Verma (2026) supply, in different settings, and that most self-repair claims do not.

11. The strongest case against this paper

11.1 "This is a dispute about a word"

The objection is that self-repair is a loose label that everyone understands loosely, and that sorting its uses is commentary. The reply is that the label is not harmless when it travels. The Hydra effect is read as redundancy that compensates for damage (Kerce, 2026), and Sections 5 and 7 show that it is evidence about one fault model that deployment does not produce. Neuromorphic self-repair is presented as a step toward platforms that repair themselves autonomously (Rastogi et al., 2021), and Section 6 shows that its largest single step is a rescaling. Code self-repair is reported as a capability, and Section 9 shows that at small scale its gain is matched by ignoring the failure, and that above 7 billion parameters the only comparison with resampling is not matched on tokens or temperature. In each case, the reading the word invites goes beyond the evidence in a direction that matters for anyone relying on it.

11.2 "The counterweight result is one preprint on one task"

This is correct, and Section 5.4 says so. The paper's argument does not depend on it alone. The decompositions of McGrath et al. (2023), McDougall et al. (2023) and Rushing and Nanda (2024) already attribute most of the identified response in two cases, and 39 percent in the third, to components that suppress or rescale in ordinary operation, which is the

Algorithm 1. Licensing a self-repair claim for a trained network

```

Input: system S (network N plus any wrapper), claimed fault class F,
       restored quantity Q, reference R, evidence E
1  if F is unstated or has no counterpart in deployment of S:
2    return "ablation-robustness of Q under F only"
3  if Q is a proxy (logit, direct effect) and the claim is about
   task function:
4    require a measurement of task function under F; else downgrade
5  split recovery in E into: scale step (renormalise, clip, LayerNorm)
   and remainder; report both
6  if the remainder is produced outside S (fine-tune, edit, reload):
7    return "external repair to R" (M4)
8  run control C: same budget, no fault (or opposite-sign dose)
9  if response under C  $\approx$  response under F:
10   return "fixed counterweight or redundancy" (M1/M2)
11 if response changes weights or policy and persists:
12   return "self-repair of S under F, measured on Q" (M3/M5)
13 return "per-input compensation under F, measured on Q"

```

counterweight reading at the level of mechanisms; Ahmad et al. (2026) add the dose-response form and a test. If their law fails on generative tasks, the Hydra effect might contain a fault-conditional part that this paper has not seen. That is a reason to run the test Section 12 proposes, not a reason to read the effect as repair in the meantime.

11.3 "Scale is function"

A reader may argue that restoring scale is restoring function, since a network whose activations are at the wrong scale computes the wrong thing, and so the distinction drawn in Section 6 is empty. The reply is that the distinction is about what is recovered, not about whether recovery helps. In Figure 1b, renormalisation restores most of the accuracy that can be restored, but at 90 percent deletion it leaves 33 points unrecovered, and the learning rule recovers 8.89 of them; presumably, content that was carried by the deleted synapses is not brought back by rescaling, and some of it is brought back by learning. The distinction tells a designer which mechanism can be expected to help under which fault - scale regulation for faults that mainly change magnitudes, plasticity or backup computation for faults that remove content - and that is the practical value of drawing it.

11.4 "The Hydra effect is still robustness, whatever it is called"

It is robustness to ablation, and Section 5.6 states what that licenses. The paper's point is that robustness to ablation is not evidence of robustness to deployment faults, and the super-weight result shows how far apart the two can be in models of the same kind. Calling the effect robustness is accurate if the fault model is attached.

11.5 "Some of these sources are not about neural networks at all"

Robots, neurons and dependable-computing taxonomies are included because they are where the strong meaning of the word comes from. The paper uses them to state what a full repair loop contains, not as evidence about transformers.

12. What would settle the open parts

Five studies would settle what the published record leaves open. Each uses existing models and existing methods.

First. Dose-response at scale and on generation. Apply the dose axis of Ahmad et al. (2026) to generative tasks and to the IOI and factual-recall settings of McGrath et al. (2023) and Rushing and Nanda (2024), at both signs of lambda. A counterweight responds linearly through zero, to doses of either sign; a fault-conditional mechanism responds to removal and not to amplification. The test distinguishes M2 from a repair.

Second. Scale-matched faults. Construct ablations that remove a component's content but preserve the residual-stream norm, for example by replacing the component's output with a random vector of the same norm in the subspace it writes to, and compare self-repair against ordinary zero, mean and resample ablation. The scale conjecture of Section 6.2 predicts that the LayerNorm share of self-repair vanishes and that total self-repair falls by roughly that share.

Third. Deployment faults in a self-repair model. In a model where the Hydra effect has been measured, inject the faults that deployment produces - random and targeted bit flips, stuck-at values, corrupted activations - in the same components, and measure whether the same downstream components compensate. This joins the interpretability and hardware literatures directly, and nobody in the sources read here has run it.

Fourth. The two-step decomposition everywhere. Report recovery as a scale step and a remainder, as Rastogi et al. (2021) did, for every neuromorphic and hardware self-repair claim, and for healing after pruning (Gromov et al., 2024). A repair claim whose remainder is near zero is a normalisation claim.

Fifth. Matched-budget control at frontier scale. Run the blind-resampling, placebo and feedback conditions of Verma (2026) and Iscan (2026) on the strongest current code models, at matched token budget and matched temperature. This is the controlled version of the repair-against-resampling comparison that Arimbur (2026) reports. If feedback beats blind resampling there,

behavioural self-repair satisfies C4 at that scale; if it does not, the reported gains are gains from attempts.

13. What is not known

It is not known whether the counterweight law holds outside the single verdict task on which it has been tested. It is not known whether transformer components that compensate for ablation would compensate for any deployment fault; no study read for this paper has checked. It is not known how much of the Hydra effect in large models is scale restoration, since the 30 percent LayerNorm estimate is described by its own authors as directional. It is not known whether astrocyte-modulated self-repair scales beyond networks of a few hundred neurons on MNIST-scale tasks, since the evaluations read for this paper stop there. It is not known whether code self-repair beats token- and temperature-matched resampling for frontier models; the one comparison above 7 billion parameters favours resampling at 8 billion parameters and repair for the three stronger models, but changes temperature along with feedback (Arimbur, 2026). And there is no shared metric for self-repair across the three literatures: the transformer studies report fractions of a direct effect, the neuromorphic studies report accuracy, and the hardware studies report failure rates, so the numbers in Table 2 cannot be pooled.

14. Limitations

This paper is the author's analysis of published work and inherits its sources' scopes. Several load-bearing results are 2025 and 2026 preprints - the counterweight law (Ahmad et al., 2026), conditional co-ablation (Gong et al., 2026), the per-layer supervision result (Kerce, 2026), the data-structure account (Rohweder et al., 2026), the vision-transformer fault study (Bhaduri et al., 2026), and the code self-repair controls (Verma, 2026; Iscan, 2026; Arimbur, 2026) - and some are single-group results not yet replicated. Several hardware and neuromorphic results are read from abstracts only, and the paper reports only what those abstracts state. The scale regularity of Section 6 is drawn from three literatures that measure different quantities; it is offered as a regularity with a proposed mechanism and a test, not as a measured law. The x positions in Figure 1a are the author's arithmetic on published counts, and the nominal parameter count used for Llama-7B is labelled as such. The paper treats unlearning and editing briefly and relies on the author's earlier papers for that literature. It does not cover self-correction of reasoning, which is a different question.

15. Conclusion

"Self-repair" names five mechanisms: scale regulation, fixed redundancy and counterweights, deviation-triggered plasticity, external repair against a reference, and behavioural search. They differ in what fault they answer, what they restore, who does the restoring, and whether they respond to the fault at all. In the

transformer case, the restoration is partial, measured under an experimenter's intervention rather than a deployment fault, carried largely by components that rescale or suppress in ordinary operation, and, on the best current evidence, not conditional on the fault. In the one neuromorphic study that separates the steps, most of the measured recovery is a rescaling, and a smaller, real part is learned. Across the literatures, networks of the kinds that absorb large structured damage can fail under small targeted damage, and the evidence is consistent with that damage striking the machinery that sets their scale. The phrase "a network that restores its own performance" can be licensed by deviation-triggered plasticity, the only mechanism inside the network whose design meets all five conditions, although no study read here yet shows all five met together; it is licensed for search-based systems at the level of the system. For the rest, the accurate description is narrower - robustness of a stated quantity under a stated fault - and the five conditions and the procedure in Algorithm 1 are offered as a way to say which description a given result has earned.

References

- Ahmad, A., Seth, P., & Sankarapu, V. K. (2026). Every Ablation Is a Dose: Counterweights and the Semblance of Self-Repair. arXiv:2610.02173. doi:10.48550/arXiv.2610.02173. [PREPRINT]
- Arimbur, J. J. (2026). How Many Tries Does It Take? Iterative Self-Repair in LLM Code Generation Across Model Scales and Benchmarks. arXiv:2604.10508. doi:10.48550/arXiv.2604.10508. [PREPRINT]
- Avizienis, A., Laprie, J.-C., Randell, B., & Landwehr, C. (2004). Basic concepts and taxonomy of dependable and secure computing. *IEEE Transactions on Dependable and Secure Computing*, 1(1). doi:10.1109/TDSC.2004.2.
- Belrose, N., Ostrovsky, I., McKinney, L., Furman, Z., Smith, L., Halawi, D., Biderman, S., & Steinhardt, J. (2023). Eliciting Latent Predictions from Transformers with the Tuned Lens. arXiv:2303.08112. doi:10.48550/arXiv.2303.08112.
- Bhaduri, P. K., Taheri, M., Nazari, S., Jenihhin, M., Herglotz, C., & Hubner, M. (2026). SENTRY: Statistical Reliability Analysis of Vision Transformers Under Soft Errors. arXiv:2606.07620. doi:10.48550/arXiv.2606.07620. [PREPRINT]
- Bongard, J., Zykov, V., & Lipson, H. (2006). Resilient Machines Through Continuous Self-Modeling. *Science*, 314(5802). doi:10.1126/science.1133687.
- Chai, D., Liu, Z., Wang, S., Pei, S., Liu, C., Li, H., & Wang, S. (2026). Analysis of LLM Vulnerability to GPU Errors: An Instruction-Level Fault Injection Study. arXiv:2601.19912. doi:10.48550/arXiv.2601.19912. [PREPRINT]
- Chen, Z., Li, G., & Pattabiraman, K. (2020). A Low-cost Fault Corrector for Deep Neural Networks through Range Restriction. arXiv:2003.13874. doi:10.48550/arXiv.2003.13874.
- Chen, Z., Li, Q., & Zhang, Z. (2022). Self-Healing Robust Neural Networks via Closed-Loop Control. arXiv:2206.12963. doi:10.48550/arXiv.2206.12963.
- Csordas, R., Manning, C. D., & Potts, C. (2025). Do Language Models Use Their Depth Efficiently? arXiv:2505.13898. doi:10.48550/arXiv.2505.13898.
- Cully, A., Clune, J., Tarapore, D., & Mouret, J.-B. (2015). Robots that can adapt like animals. *Nature*, 521. doi:10.1038/nature14422. arXiv:1407.3501. doi:10.48550/arXiv.1407.3501.
- Deeb, A., & Roger, F. (2024). Do Unlearning Methods Remove Information from Language Model Weights? arXiv:2410.08827. doi:10.48550/arXiv.2410.08827.

- Dixit, H. D., Pendharkar, S., Beadon, M., Mason, C., Chakravarthy, T., Muthiah, B., & Sankar, S. (2021). Silent Data Corruptions at Scale. *arXiv:2102.11245*. doi:10.48550/arXiv.2102.11245.
- Edelman, G. M., & Gally, J. A. (2001). Degeneracy and complexity in biological systems. *Proceedings of the National Academy of Sciences*, 98(24). doi:10.1073/pnas.231499798.
- Elhoushi, M., Shrivastava, A., Liskovich, D., Hosmer, B., Wasti, B., Lai, L., et al. (2024). LayerSkip: Enabling Early Exit Inference and Self-Speculative Decoding. *arXiv:2404.16710*. doi:10.48550/arXiv.2404.16710.
- Fan, A., Grave, E., & Joulin, A. (2019). Reducing Transformer Depth on Demand with Structured Dropout. *arXiv:1909.11556*. doi:10.48550/arXiv.1909.11556.
- Frantar, E., & Alistarh, D. (2023). SparseGPT: Massive Language Models Can Be Accurately Pruned in One-Shot. *arXiv:2301.00774*. doi:10.48550/arXiv.2301.00774.
- Gong, Z., Lu, H., Wang, T., Zhang, Y., Wang, Y., Zeng, Z., Xiao, M., Yuen, C., & Lim, W. Y. B. (2026). Conditional Co-Ablation: Recovering Self-Repair Backups in Transformer Circuits. *arXiv:2607.01940v3*. doi:10.48550/arXiv.2607.01940. [PREPRINT]
- Gromov, A., Tirumala, K., Shapourian, H., Glorioso, P., & Roberts, D. A. (2024). The Unreasonable Ineffectiveness of the Deeper Layers. *ICLR 2025*. *arXiv:2403.17887*. doi:10.48550/arXiv.2403.17887.
- Guo, J., Chakrabarti, C., & Fan, D. (2025). SBFA: Single Sneaky Bit Flip Attack to Break Large Language Models. *arXiv:2509.21843*. doi:10.48550/arXiv.2509.21843. [PREPRINT]
- Gupta, A., Rao, A., & Anumanchipalli, G. (2024). Model Editing at Scale leads to Gradual and Catastrophic Forgetting. *arXiv:2401.07453*. doi:10.48550/arXiv.2401.07453.
- Han, Z., Islam, A. N. M. N., & Sengupta, A. (2022). Astromorphic Self-Repair of Neuromorphic Hardware Systems. *arXiv:2209.07428*. doi:10.48550/arXiv.2209.07428.
- Hase, P., Bansal, M., Kim, B., & Ghandeharioun, A. (2023). Does Localization Inform Editing? Surprising Differences in Causality-Based Localization vs. Knowledge Editing in Language Models. *arXiv:2301.04213*. doi:10.48550/arXiv.2301.04213.
- Hassibi, B., Stork, D. G., & Wolff, G. J. (1993). Optimal Brain Surgeon and general network pruning. *IEEE International Conference on Neural Networks*. doi:10.1109/ICNN.1993.298572.
- He, S., Sun, G., Shen, Z., & Li, A. (2024). What Matters in Transformers? Not All Attention is Needed. *arXiv:2406.15786*. doi:10.48550/arXiv.2406.15786.
- Heimersheim, S., & Nanda, N. (2024). How to use and interpret activation patching. *arXiv:2404.15255*. doi:10.48550/arXiv.2404.15255.
- Hoang, L.-H., Hanif, M. A., & Shafique, M. (2020). FT-ClipAct: Resilience Analysis of Deep Neural Networks and Improving their Fault Tolerance using Clipped Activation. *DATE 2020*. *arXiv:1912.00941*. doi:10.48550/arXiv.1912.00941.
- Hong, S., Frigo, P., Kaya, Y., Giuffrida, C., & Dumitras, T. (2019). Terminal Brain Damage: Exposing the Graceless Degradation in Deep Neural Networks Under Hardware Fault Attacks. *USENIX Security 2019*. *arXiv:1906.01017*. doi:10.48550/arXiv.1906.01017.
- Hu, S., Fu, Y., Wu, Z. S., & Smith, V. (2024). Unlearning or Obfuscating? Jogging the Memory of Unlearned LLMs via Benign Relearning. *ICLR 2025*. *arXiv:2406.13356*. doi:10.48550/arXiv.2406.13356.
- Iscan, M. (2026). Falsification, Not Exposure: An Internally Preregistered Placebo-Controlled Decomposition of Self-Repair Feedback in Frozen Small Code Models. *arXiv:2606.31511*. doi:10.48550/arXiv.2606.31511. [PREPRINT]
- Jastrzebski, S., Arpit, D., Ballas, N., Verma, V., Che, T., & Bengio, Y. (2017). Residual Connections Encourage Iterative Inference. *arXiv:1710.04773*. doi:10.48550/arXiv.1710.04773.
- Jonas, E., & Kording, K. P. (2017). Could a Neuroscientist Understand a Microprocessor? *PLOS Computational Biology*, 13(1). doi:10.1371/journal.pcbi.1005268.
- Karamat, M. Z., Saif, S., & Garcia, C. C. (2025). BitFlipScope: Scalable Fault Localization and Recovery for Bit-Flip Corruptions in LLMs. *arXiv:2512.22174*. doi:10.48550/arXiv.2512.22174. [PREPRINT]
- Keinan, A., Sandbank, B., Hilgetag, C. C., Meilijson, I., & Ruppin, E. (2004). Fair Attribution of Functional Contribution in Artificial and Biological Networks. *Neural Computation*, 16(9). doi:10.1162/0899766041336387.
- Kephart, J. O., & Chess, D. M. (2003). The vision of autonomic computing. *Computer*, 36(1). doi:10.1109/MC.2003.1160055.
- Kerce, J. C. (2026). Engineering Verifiable Modularity in Transformers via Per-Layer Supervision. *arXiv:2603.18029*. doi:10.48550/arXiv.2603.18029. [PREPRINT]
- Lad, V., Lee, J. H., Gurnee, W., & Tegmark, M. (2024). The Remarkable Robustness of LLMs: Stages of Inference? *arXiv:2406.19384*. doi:10.48550/arXiv.2406.19384.
- Li, G., Hari, S. K. S., Sullivan, M. B., Tsai, T. K., Pattabiraman, K., Emer, J., & Keckler, S. W. (2017). Understanding error propagation in deep learning neural network (DNN) accelerators and applications. *SC 2017*. doi:10.1145/3126908.3126964.
- Li, M., & Janson, L. (2024). Optimal ablation for interpretability. *arXiv:2409.09951*. doi:10.48550/arXiv.2409.09951.
- Liu, J., Harkin, J., Maguire, L. P., McDaid, L. J., & Wade, J. J. (2018). SPANNER: A Self-Repairing Spiking Neural Network Hardware Architecture. *IEEE Transactions on Neural Networks and Learning Systems*, 29(4), 1287-1300. doi:10.1109/TNNLS.2017.2673021.
- Lynch, A., Guo, P., Ewart, A., Casper, S., & Hadfield-Menell, D. (2024). Eight Methods to Evaluate Robust Unlearning in LLMs. *arXiv:2402.16835*. doi:10.48550/arXiv.2402.16835.
- Ma, J., Pei, H., Lausen, L., & Karypis, G. (2025). Understanding Silent Data Corruption in LLM Training. *arXiv:2502.12340*. doi:10.48550/arXiv.2502.12340.
- Mahendrakar, P. (2026a). Three Guarantees Under One Word: What Importance-Based Selective Forgetting in Language Models Can and Cannot Promise. *Life of Research*. doi:10.5281/zenodo.22054075.
- Mahendrakar, P. (2026b). Four Things Called Forgetting: Interference, Transience, Reset and Unlearning Remove Different Things, Why Forgetting Looks Easy by Accident and Hard on Purpose Only Under Different Instruments, and the Savings Measurement That Would Tell Them Apart. *Life of Research*. doi:10.5281/zenodo.23004075.
- McDougall, C., Conmy, A., Rushing, C., McGrath, T., & Nanda, N. (2023). Copy Suppression: Comprehensively Understanding an Attention Head. *arXiv:2310.04625*. doi:10.48550/arXiv.2310.04625.
- McGrath, T., Rahtz, M., Kramar, J., Mikulik, V., & Legg, S. (2023). The Hydra Effect: Emergent Self-repair in Language Model Computations. *arXiv:2307.15771*. doi:10.48550/arXiv.2307.15771.
- Men, X., Xu, M., Zhang, Q., Wang, B., Lin, H., Lu, Y., Han, X., & Chen, W. (2024). ShortGPT: Layers in Large Language Models are More Redundant Than You Expect. *arXiv:2403.03853*. doi:10.48550/arXiv.2403.03853.
- Meng, K., Bau, D., Andonian, A., & Belinkov, Y. (2022). Locating and Editing Factual Associations in GPT. *arXiv:2202.05262*. doi:10.48550/arXiv.2202.05262.
- Michel, P., Levy, O., & Neubig, G. (2019). Are Sixteen Heads Really Better than One? *NeurIPS 2019*. *arXiv:1905.10650*. doi:10.48550/arXiv.1905.10650.
- Miller, J., Chughtai, B., & Saunders, W. (2024). Transformer Circuit Faithfulness Metrics are not Robust. *arXiv:2407.08734*. doi:10.48550/arXiv.2407.08734.
- Olausson, T. X., Inala, J. P., Wang, C., Gao, J., & Solar-Lezama, A. (2023). Is Self-Repair a Silver Bullet for Code Generation? *arXiv:2306.09896*. doi:10.48550/arXiv.2306.09896.

- Press, O., Schneider, S., Kummerer, M., & Bethge, M. (2023). RDumb: A simple approach that questions our progress in continual test-time adaptation. arXiv:2306.05401. doi:10.48550/arXiv.2306.05401.
- Rakin, A. S., He, Z., & Fan, D. (2019). Bit-Flip Attack: Crushing Neural Network with Progressive Bit Search. arXiv:1903.12269. doi:10.48550/arXiv.1903.12269.
- Rastogi, M., Lu, S., Islam, N., & Sengupta, A. (2021). On the Self-Repair Role of Astrocytes in STDP Enabled Unsupervised SNNs. *Frontiers in Neuroscience*, 14. doi:10.3389/fnins.2020.603796. arXiv:2009.03473. doi:10.48550/arXiv.2009.03473.
- Reagen, B., Gupta, U., Pentecost, L., Whatmough, P., Lee, S. K., Mulholland, N., Brooks, D., & Wei, G.-Y. (2018). Ares: A framework for quantifying the resilience of deep neural networks. *Proceedings of the 55th Annual Design Automation Conference*. doi:10.1145/3195970.3195997.
- Rohweder, J., Dutta, S., & Gurevych, I. (2026). Hierarchical Latent Structures in Data Generation Process Unify Mechanistic Phenomena across Scale. arXiv:2603.06592. doi:10.48550/arXiv.2603.06592. [PREPRINT]
- Rushing, C., & Nanda, N. (2024). Explorations of Self-Repair in Language Models. *ICML 2024*. arXiv:2402.15390. doi:10.48550/arXiv.2402.15390.
- Sotoudeh, M., & Thakur, A. V. (2021). Provable Repair of Deep Neural Networks. *PLDI 2021*. arXiv:2104.04413. doi:10.48550/arXiv.2104.04413.
- Stolfo, A., Wu, B., Gurnee, W., Belinkov, Y., Song, X., Sachan, M., & Nanda, N. (2024). Confidence Regulation Neurons in Language Models. arXiv:2406.16254. doi:10.48550/arXiv.2406.16254.
- Sun, M., Chen, X., Kolter, J. Z., & Liu, Z. (2024). Massive Activations in Large Language Models. arXiv:2402.17762. doi:10.48550/arXiv.2402.17762.
- Sun, M., Liu, Z., Bair, A., & Kolter, J. Z. (2023). A Simple and Effective Pruning Approach for Large Language Models. arXiv:2306.11695. doi:10.48550/arXiv.2306.11695.
- Sun, Q., Pickett, M., Nain, A. K., & Jones, L. (2024). Transformer Layers as Painters. arXiv:2407.09298. doi:10.48550/arXiv.2407.09298.
- Tahmasivand, A., Zahran, N., Al-Sayouri, S., Fouda, M., & Khasawneh, K. N. (2025). LM-Fix: Lightweight Bit-Flip Detection and Rapid Recovery Framework for Language Models. *IEEE ICCD 2025*. arXiv:2511.02866. doi:10.48550/arXiv.2511.02866.
- Torres-Huitzil, C., & Girau, B. (2017). Fault and Error Tolerance in Neural Networks: A Review. *IEEE Access*, 5, 17322-17341. doi:10.1109/ACCESS.2017.2742698.
- Turrigiano, G. G. (2008). The Self-Tuning Neuron: Synaptic Scaling of Excitatory Synapses. *Cell*, 135(3). doi:10.1016/j.cell.2008.10.008.
- Veit, A., Wilber, M., & Belongie, S. (2016). Residual Networks Behave Like Ensembles of Relatively Shallow Networks. *NIPS 2016*. arXiv:1605.06431. doi:10.48550/arXiv.1605.06431.
- Verma, Y. (2026). Try Again, Don't Look Back: Blind Resampling Outperforms Self-Repair in Small Code Models. arXiv:2607.26117. doi:10.48550/arXiv.2607.26117. [PREPRINT]
- Voita, E., Talbot, D., Moiseev, F., Sennrich, R., & Titov, I. (2019). Analyzing Multi-Head Self-Attention: Specialized Heads Do the Heavy Lifting, the Rest Can Be Pruned. *ACL 2019*. arXiv:1905.09418. doi:10.48550/arXiv.1905.09418.
- von Neumann, J. (1956). Probabilistic Logics and the Synthesis of Reliable Organisms From Unreliable Components. In *Automata Studies (AM-34)*. Princeton University Press. doi:10.1515/9781400882618-003.
- Wade, J. J., McDaid, L. J., Harkin, J., Crunelli, V., & Kelso, J. A. S. (2012). Self-repair in a bidirectionally coupled astrocyte-neuron (AN) system based on retrograde signaling. *Frontiers in Computational Neuroscience*, 6. doi:10.3389/fncom.2012.00076.
- Wang, D., Shelhamer, E., Liu, S., Olshausen, B., & Darrell, T. (2021). Tent: Fully Test-time Adaptation by Entropy Minimization. *ICLR 2021*. arXiv:2006.10726. doi:10.48550/arXiv.2006.10726.
- Wang, K., Variengien, A., Conmy, A., Shlegeris, B., & Steinhardt, J. (2022). Interpretability in the Wild: a Circuit for Indirect Object Identification in GPT-2 small. arXiv:2211.00593. doi:10.48550/arXiv.2211.00593.
- Yan, Y., Lu, S., Gao, Y., Li, Z., Zhao, Z., Yuan, Q., & Wang, Y. (2025). Has the Two-Decade-Old Prophecy Come True? Artificial Bad Intelligence Triggered by Merely a Single-Bit Flip in Large Language Models. arXiv:2510.00490. doi:10.48550/arXiv.2510.00490. [PREPRINT]
- Yu, M., Wang, D., Shan, Q., Reed, C. J., & Wan, A. (2024). The Super Weight in Large Language Models. arXiv:2411.07191. doi:10.48550/arXiv.2411.07191.
- Zhang, F., & Nanda, N. (2023). Towards Best Practices of Activation Patching in Language Models: Metrics and Methods. arXiv:2309.16042. doi:10.48550/arXiv.2309.16042.