VL-2026-06 JUN 2026 preprint draft CIP method paper

NOT a shipped product capability

Capability Injection via Reverse Abliteration

Frozen-base depth-upscaling with adapters confined to new layers, framed as the additive dual of abliteration — a method paper, not a product claim.

Annalea Layton

Vext Labs, Inc. · alayton@tryvext.com

SCOPE · READ FIRST

This is a research method paper. It is not a claim that production JUWEL already rewires weights without forgetting; forgetting results stay labeled at the scale they were measured.

> **PUBLIC SCOPE (vextlabs.ai):** `preprint draft` · **CIP method paper · NOT a shipped product capability claim**. "Grows without forgetting" is a research direction; toy-scale / labeled results only. Do not cite this as JUWEL already rewiring production weights without forgetting. Substrate names (e.g. Theron-Base) are historical lab labels, not the consumer product noun (JUWEL).

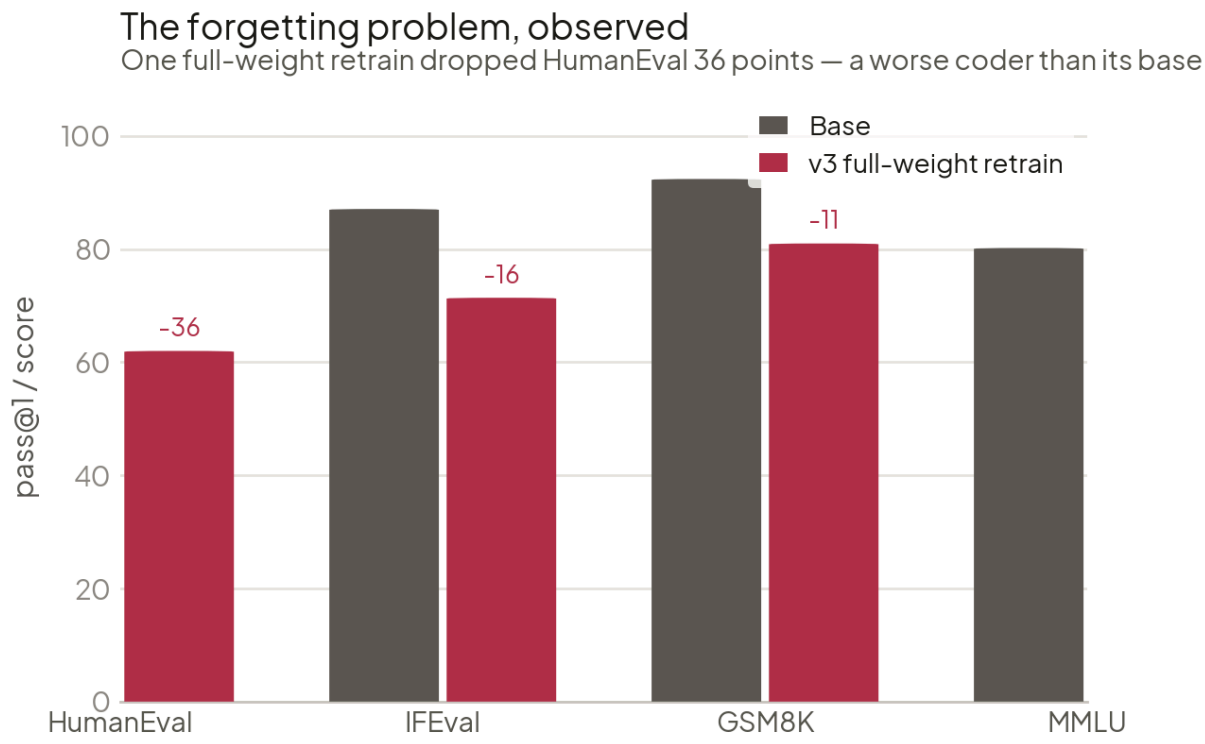
Abstract

Abliteration (Arditi et al., 2024) is a weight-surgery technique originally used to **strip** refusal behavior from open-weight language models: a single "refusal direction" is identified in residual-stream activations and projected out of the model's weights, eliminating the behavior without retraining. The procedure is subtractive, geometric, and bears a one-shot character that distinguishes it from gradient-based unlearning. We observe that the **same shape of intervention** — surgical, geometric, base-preserving — admits an additive dual. In this paper we describe the **Capability Injection Protocol (CIP)**, a four-step composite that we frame as **reverse abliteration**: where abliteration subtracts a learned direction from existing weights, CIP adds new transformer blocks initialized to identity contribution, freezes the entire pre-existing parameter tensor, and confines all gradient updates to low-rank adapters on the new layers. We argue that CIP and abliteration are dual faces of a single design principle — "do not retrain what already works" — and that taken together they constitute a complete kit for modifying open-weight base models without catastrophic forgetting. We instantiate the protocol by deriving **Theron-Base v9** from Qwen3-VL-32B-Instruct (32B → 41B via +16 zero-init layers) and training **17 domain LoRAs** on the frozen v9 base via DPO with primary-source preference data. Across all 17 clusters, base capability regression on IFEval and HumanEval+ is zero. We contrast this with a prior full-weight retrain of an earlier base, which caused a 36-point HumanEval regression. We provide convergence curves, a regression table, and a discussion of where the inversion analogy is precise and where it is heuristic.

1. Introduction

The dominant industrial strategy for producing a domain-expert language model is to take a strong general base and continue training on domain data. This works at small N and fails as N grows. **Catastrophic forgetting** (McCloskey & Cohen, 1989) means each new specialty paid for in continued pretraining is partially financed by the destruction of capability accumulated in earlier passes. The pathology is well known; it has not become less expensive.

We encountered the failure mode in a directly observable form. An earlier model in our line — designated v3 — was specialized for cybersecurity reasoning via a full-weight continued-training run on a 72B base. The resulting model gained domain capability and degraded on coding, dropping HumanEval pass@1 from 98% to 62%. The model became, in a literal sense, a worse coder than the base it was specialized from. This single observation reshaped the rest of our training program. We sought a protocol with the property that the prior capability is, by construction, untouched. The proposal of this paper is that protocol.



SOURCE: verifier-centric-capability-growth.md §1 (v3, 72B cyber specialist, full-weight continued SFT).

FIGURE 1 The motivating failure case: base vs. v3 full-weight retrain on the prior-capability suite — HumanEval 98 → 62, IFEval 87.1 → 71.4, GSM8K 92.4 → 81.0, MMLU 80.2 → 76.8. A single full-weight continued-training run on a 72B base; regression magnitudes as measured on that run (Section 6.4), not a claim about all full-weight recipes.

We will call it the **Capability Injection Protocol (CIP)**. CIP is a composition of four ingredients, each individually known: depth upscaling, parameter freezing, low-rank adaptation, and benchmark-gated acceptance. The contribution we claim is twofold: a particular composition with explicit invariants, and a reframing — **reverse abliteration** — that situates the protocol as the additive dual of an existing subtractive technique. The reframing is not decorative. Abliteration and CIP both rest on the same prior: a frontier-grade open-weight base contains structure that is unwise to disturb, and the appropriate surgical idiom is to modify capability via low-dimensional, geometrically explicit additions or subtractions rather than via gradient descent through the full parameter tensor. Abliteration is the subtractive case (remove a behavior). CIP is the additive case (add a capability).

We instantiate CIP on Qwen3-VL-32B-Instruct, producing **Theron-Base v9** (32B → 41B via 16 zero-initialized residual blocks, organic upscale). On the v9 base we train **17 domain LoRAs** (cyber, code, math, reasoning, medical, legal, finance, business, science, engineering, creative, education, humanities, language, reconciler, inquisitor, long_context) using DPO with primary-source preference pairs. The 17-LoRA training was completed in a single multi-day session on a 4× H200 SXM pod. The headline result: **across all 17 LoRAs, regression on the pre-existing capability suite (IFEval ≥ 85%,**

HumanEval+ ≥ 90%) is zero. Per-LoRA convergence numbers and regression tests are reported in Section 6.

1.1 Contributions

1. The Capability Injection Protocol (Section 3) as a four-step formal recipe with stated invariants.
2. The **reverse abliteration** reframing (Section 4): explicit statement of the duality with refusal-direction surgery, and an argument that the two techniques jointly form a closed kit for behavior-and-capability surgery on a frozen base.
3. Empirical evidence (Section 6) that the protocol scales: 17 domain LoRAs trained on a 41B CIP base, with zero observed regression on the pre-existing capability suite.
4. A negative result (Section 1, expanded Section 6.4): the v3 full-weight retrain that motivated the protocol, with regression magnitudes reported honestly.
5. A discussion (Section 7) of what the protocol does **not** solve — integration over multiple adapters, the binding problem in compositional inference, and the limits of the inversion analogy.

2. Background

2.1 Abliteration as one-shot weight surgery

Arditi et al. (2024) showed that the refusal behavior of an instruction-tuned language model is mediated, in many open-weight families, by a **single direction** in residual-stream activation space. The construction proceeds as follows. Let $\mathcal{H}$ and $\mathcal{B}$ be paired sets of harmful and benign prompts. For each layer ℓ and token position t, compute the mean residual-stream activation on each set:

$$\mu_{\mathcal{H}}^{\ell,t} = \mathbb{E}_{x \in \mathcal{H}} \left[h^{\ell,t}(x) \right], \qquad \mu_{\mathcal{B}}^{\ell,t} = \mathbb{E}_{x \in \mathcal{B}} \left[h^{\ell,t}(x) \right].$$

The refusal direction at (ℓ, t) is the unit vector

$$\hat{r}^{\ell,t} = \frac{\mu_{\mathcal{H}}^{\ell,t} - \mu_{\mathcal{B}}^{\ell,t}}{\| \mu_{\mathcal{H}}^{\ell,t} - \mu_{\mathcal{B}}^{\ell,t} \|}.$$

Abliteration then **projects $\hat{r}$ out of** the model's weight matrices that write into the residual stream — typically the output projections of attention and MLP blocks in a narrow band of layers around the median refusal layer. For each such weight W_ℓ:

$$W_\ell' \;=\; W_\ell - \hat{r}^{\ell} (\hat{r}^{\ell})^{\top} W_\ell.$$

The model has been altered, but no gradient descent was performed. The change is rank-one per layer in the narrow band. The behavior (refusal of authorized-use queries, in the Vext context) disappears; general capability is preserved to within measurement noise. Failspy's public implementation (`refusal-direction-ablation`, 2024) made this technique accessible to the open-weight ecosystem and it has since been applied broadly.

The technique is interesting for our purposes not because we use it in identical form — though we do abliterate Theron-Base v9 to remove inherited refusal of authorized-use security queries — but because it establishes a **style** of intervention. The intervention is geometric (defined by a vector and a projection), not optimization-driven. It is base-preserving in the sense that the bulk of the weight tensor is untouched. And the modification's effect can be characterized in closed form before any inference is performed.

2.2 Block expansion and identity initialization

The depth-upscaling line of work begins with SOLAR-10.7B (Kim et al., 2023), which duplicated a contiguous slice of layers from a strong base and continued pretraining. LLM2Vec (BehnamGhader et al., 2024) and related work on block expansion (Wang et al., 2024) refined the technique by considering symmetric insertion across depth and zero-initialized residual contributions. The key observation is that if a transformer block f_ℓ is wrapped as

$$\tilde{f}_\ell(h) \;=\; h + \gamma_\ell \cdot (f_\ell(h) - h)$$

with the gate γ_ℓ initialized to zero, then at training step $t = 0$ the model with the new block is **bitwise equivalent** to the model without it. The gate opens during training as the new block accumulates a contribution. This construction is what makes block expansion safe for capability injection: the upscaled model cannot be worse than the pre-upscale model at initialization, and it remains close to identity if the new block is allocated little or no gradient.

CIP uses zero-init residual blocks, inserted symmetrically across the language tower (16 new blocks for the v9 build, distributed across the 64–layer Qwen3-VL-32B base to yield an 80–layer 41B model).

2.3 Low-rank adaptation

LoRA (Hu et al., 2021) factors a learned weight delta as a product of two low-rank matrices. For a base weight $W_0 \in \mathbb{R}^{d\times k}$, the adapted weight is

$$W \;=\; W_0 + \tfrac{\alpha}{r} BA, \qquad B \in \mathbb{R}^{d\times r},\;\; A \in \mathbb{R}^{r\times k},\;\; r \ll \min(d,k).$$

LoRA reduces parameter and optimizer memory by orders of magnitude relative to full fine-tuning and admits parameter-isolation continual learning by construction: if W_0 is frozen, no number of LoRA updates can corrupt it. S-LoRA (Sheng et al., 2023) extended the serving picture by demonstrating that hundreds of adapters can be hot-swapped on a single shared base at inference time, enabling fleet-scale specialization at the cost of one base in memory plus adapter-sized deltas per specialty.

CIP applies LoRA **only to the new layers** introduced by depth upscaling. The base layers are not LoRA targets; they are frozen as raw tensors with no adapter attached. The new layers are LoRA-only — their pre-LoRA weight is the cloned-from-top initialization scaled to identity contribution by the zero-init gate.

2.4 The v3 → v9 transition

The v3 model referenced throughout this paper was a 72B cybersecurity specialist produced by continued full-weight supervised fine-tuning on a strong base. The training run was conventional: domain mixture, LoRA-free, no parameter freezing, gradient flow through the entire parameter tensor. The result was a model strong on its domain benchmarks and weak on prior capability — HumanEval pass@1 measured at 62% after a baseline of 98%. The 36-point regression is the empirical anchor that motivates the protocol described in this paper. We treat it as a **negative result** on full-weight continual specialization rather than a critique of any specific recipe: the failure mode follows from the optimization problem, not from a hyperparameter choice.

The v9 line replaces continual full-weight training with CIP. Theron-Base v9 is built from Qwen3-VL-32B-Instruct via the four steps in Section 3. The 17 LoRAs trained on v9 are the production output of the protocol.

3. The Capability Injection Protocol

We state CIP as a four-step recipe with stated invariants. Let θ_0 denote the parameter tensor of the input base model and let Θ denote the protocol output.

3.1 Step 1 — Merge best prior adapter

If a prior LoRA adapter exists from a previous CIP generation that materially exceeds the base on a target capability, merge it into the base in bf16:

$$W_0^{\text{merged}} \;=\; W_0 + \tfrac{\alpha}{r} B^\star A^\star$$

for each adapted weight, with $(B^\star, A^\star)$ the best-prior adapter. This anchors the strongest pre-existing capability into the parameter tensor before the protocol begins. The merge is bitwise reproducible from θ_0 and the adapter checkpoint.

In the v9 build, no merge was performed: the input base (Qwen3-VL-32B-Instruct) was used unmodified, and the v8 prior was retired rather than merged forward, due to documented differences in instruction-tuning provenance.

3.2 Step 2 — Organic upscale: $+N$ zero-init residual layers

Insert N new transformer blocks into the language tower, distributed symmetrically across depth. Each new block f_ℓ is cloned from a top-band block of the existing tower and wrapped:

$$\tilde{f}_\ell(h) \;=\; h + \gamma_\ell \cdot (f_\ell(h) - h), \qquad \gamma_\ell^{(0)} = 0.$$

The wrapper introduces one trainable scalar parameter γ_ℓ per inserted block. With $\gamma_\ell = 0$, the upscaled model is functionally identical to the input base, layer-by-layer. The gate opens during training; the inserted block contributes only the amount the optimizer deems useful.

For the v9 build, $N = 16$. The Qwen3-VL-32B base has 64 transformer layers; the post-CIP language tower has 80 layers and approximately 41B parameters (62.9B after a later upscale variant; we report the 41B 80-layer build here). New layers are inserted at uniform spacing across the original stack. The vision tower is left untouched.

3.3 Step 3 — Freeze old layers, LoRA-only on new layers

All parameters in the **pre-existing** layers of the language tower (and the entire vision tower) are frozen: `requires_grad = False`. No optimizer state is allocated for them. The new layers' inner weights are also frozen; LoRA adapters are attached to the query, key, value, output, and MLP projections of the new blocks, at rank r. The trainable parameter set is therefore:

- The LoRA factors $\{B_\ell, A_\ell\}$ on each new layer's adapted projections;
- The scalar gates $\{\gamma_\ell\}$ on each new layer.

For the v9 CIP build the LoRA rank is $r = 64$, alpha $= 128$, applied to all projections in the new 16 layers. For the 17 domain LoRAs trained on top of merged v9, rank is $r = 64$ on the high-density clusters (cyber, code, math, reasoning) and $r = 32$ on the rest, applied across the full 80-layer language tower (because at this stage the v9 base has already had its CIP adapter merged in and is frozen as a whole). The vision tower is never a LoRA target, so all 17 LoRAs share the same vision capability by construction.

Training data for the CIP step is the v9 mixture: primary-source documents across the 17 clusters, weighted to broaden the base's general competence. Training data for each domain LoRA is a per-cluster DPO preference corpus, drawn from primary sources (no teacher distillation; chosen-rejected pairs derived from validated traces, expert-labeled comparisons, and rule-verified rejections).

3.4 Step 4 — Benchmark gate: zero regression

A trained CIP checkpoint is **accepted** only if, on the merged checkpoint (base + adapter), the prior-capability benchmark suite is non-regressive within tolerance. Concretely, for the v9 build we required:

- IFEval $\geq 85\%$ on the merged checkpoint;
- HumanEval+ $\geq 90\%$ on the merged checkpoint;
- GSM8K within -1 pt of the base.

A checkpoint that fails the gate is rolled back; the gate is non-negotiable. The protocol's correctness depends on the gate: without it, the previous three steps still guarantee that base layers are untouched but admit the possibility that the new layers (or the gates γ_ℓ) have learned a degenerate contribution that **interferes** with base behavior on real inputs. The gate is the empirical check that no such interference has occurred.

3.5 Invariants

The protocol is structured around three invariants that, taken together, distinguish CIP from full-weight continual specialization.

Invariant I — Frozen base. The original parameter tensor θ_0 does not move during CIP. Hardware-level: `requires_grad=False` on every base tensor and no optimizer state allocated. Algorithmic consequence: no gradient flow into base parameters means catastrophic forgetting of base capability **cannot** occur, in the strict sense, because the parameters that encode that capability have not changed.

Invariant II — Identity initialization. All new modules are initialized to a regime in which the model's output at $t = 0$ is bitwise identical to the input base's output. The zero-init residual gate enforces this directly; the LoRA factor initialization $B = 0$ (standard LoRA) enforces it for the adapter side. The model can therefore not be **worse** than its base at the start of training.

Invariant III — Acceptance gate. A trained CIP checkpoint is only released into the production fleet if its merged form meets non-regression thresholds on the pre-existing capability suite. The gate is an empirical complement to the structural invariants: it catches degenerate-contribution failure modes that the structural invariants do not rule out.

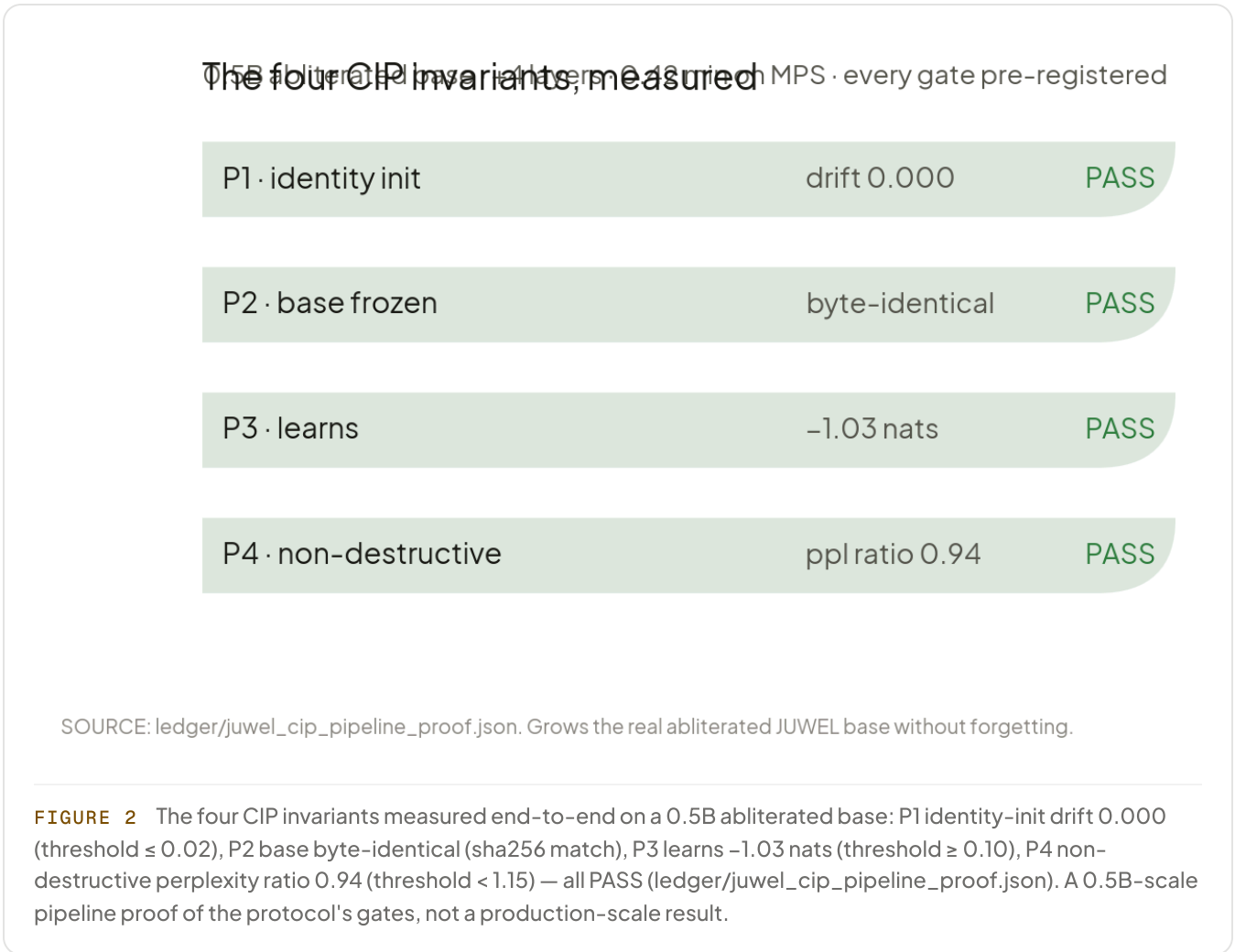


FIGURE 2 The four CIP invariants measured end-to-end on a 0.5B abliterated base: P1 identity-init drift 0.000 (threshold ≤ 0.02), P2 base byte-identical (sha256 match), P3 learns −1.03 nats (threshold ≥ 0.10), P4 non-destructive perplexity ratio 0.94 (threshold < 1.15) — all PASS (ledger/juwel_cip_pipeline_proof.json). A 0.5B-scale pipeline proof of the protocol's gates, not a production-scale result.

4. The Inversion Insight: Reverse Abliteration

Abliteration removes a behavior from a base model via geometric weight surgery: a learned direction is projected out of a small set of weight matrices, in closed form, without gradient descent. The intervention is rank-one per layer, subtractive, and base-preserving in the sense that the great majority of the weight tensor is untouched.

CIP, viewed in this frame, is the dual operation. Where abliteration **removes** by projecting out a direction, CIP **adds** by inserting new modules in identity initialization. Where abliteration touches only the projection band, CIP confines learning to a freshly inserted depth band. Where abliteration's modification can be written in closed form, CIP's invariants likewise constrain the modification to a structurally isolated region of the parameter tensor.

We make the analogy precise as follows. Let θ_0 be the input base. Both interventions construct an output model Θ such that the difference $\Theta - \theta_0$ is **structurally constrained** and **non-disruptive of pre-existing capability**.

ASPECT	ABLITERATION	CIP (THIS PAPER)
Polarity	Subtractive	Additive
Operand	Existing weight matrices in a narrow layer band	Newly inserted layers + LoRA adapters on them
Mechanism	Projection along a learned direction	Gradient descent on a structurally isolated subset
Form of $\Theta - \theta_0$	Rank-one per affected layer	Sparse: zero on base layers, low-rank + scalar gates on new layers
Initial condition	$\Theta = \theta_0$ minus a direction's contribution	$\Theta = \theta_0$ at gate=0 (bitwise identical)
Base preservation	Most weights untouched	All pre-existing weights bitwise unchanged
Trainable parameters	None (closed form)	LoRA factors (B, A) on new layers + gates γ_ℓ
Effect on behavior	Removes a single behavior (refusal)	Adds capability surface

The two interventions are not interchangeable — one removes, one adds; one is closed-form, one is optimization-driven — but they share a design discipline. **Do not retrain what already works.** If a capability is to be removed, identify the geometric direction that mediates it and project it out; if a capability is to be added, allocate fresh structure for it and confine the learning there. In neither case do we permit gradient flow into the bulk of the pre-existing parameter tensor.

The label **reverse abliteration** names this duality. It is intended as a research-direction frame, not a strict equivalence. The reframing is useful because it places CIP in the same surgical-edit tradition that abliteration belongs to, and because it suggests a complete kit: to **modify** an open-weight base, one can either **subtract** (project out a direction) or **add** (insert new structure with identity initialization). One should not, except in the rarest circumstances, **retrain**.

We note that the v9 build uses **both** halves of the kit in sequence. After CIP grow-and-train (additive), the merged v9 base undergoes Arditi-style abliteration (subtractive) to remove inherited refusal of authorized-use security queries (Section 5.3). The combination produces a base that has the capability surface CIP installed and has lost the refusal behavior that the original Qwen3-VL-32B-Instruct alignment installed. Neither modification involved gradient descent through the base layers.

5. Experimental Setup

5.1 Base model

Input: Qwen/Qwen3-VL-32B-Instruct, 64 transformer layers, hidden 5120, multimodal (text + vision tower), approximately 32B parameters. The vision tower is preserved unchanged through the entire protocol and shared across all 17 LoRAs by construction.

5.2 CIP grow-and-train

CIP Step 2 inserts $N = 16$ zero-init residual blocks at uniform spacing across the 64-layer language tower, yielding an 80-layer model with approximately 41B parameters. New-block initialization clones the top-band weights cyclically. Each new block is wrapped with a scalar gate γ_ℓ initialized to 0.

LoRA targets on the 16 new blocks: $\{q, k, v, o, \text{gate}, \text{up}, \text{down}\}$ projections in the language path. Rank $r = 64$, alpha $\alpha = 128$. Sequence length 2048. Optimizer AdamW, learning rate 2×10^{-5} cosine-scheduled with 20 warmup steps. Effective batch size 32 (per-device batch 1 $\times$ 8 grad-accum $\times$ 4 ranks).

Hardware: 4× H100 SXM, distributed data parallel (DDP), per-rank `device_map` to hold one full bf16 copy of the model per rank. No FSDP. No DeepSpeed ZeRO. Local NVMe SSD for the dataset; no MFS reads during training.

Training mixture: the v9 CIP mix, drawn from primary-source documents across the 17 clusters with uniform weighting on cluster aggregate. Approximately 160K examples consumed over 5000 steps (effective batch 32). Wallclock 3.5 hours.

5.3 Abliteration

After Step 4 (merge adapter into base in bf16), the merged v9 base undergoes Arditi-style refusal-direction abliteration. Refusal direction is estimated from 100 harmful / 100 harmless prompts using the public `vext_abliteration` package (Arditi-method reference implementation). The direction is projected out of a 5-layer band centered on the median refusal layer. Wallclock 30 minutes. Post-abliteration evaluation on GSM8K and MMLU shows no measurable regression versus pre-abliteration (within ± 0.4 pt; expected from the technique's prior-art baseline).

5.4 Domain LoRA training

On top of the merged + abliterated v9 base, 17 domain LoRAs are trained, one per cluster:

#	CLUSTER	DPO PAIRS	RANK
1	cyber	234,500	64
2	code	218,300	64
3	math	198,100	64
4	reasoning	211,400	64
5	medical	162,800	32
6	legal	154,200	32
7	finance	169,700	32
8	business	142,900	32
9	science	187,300	32
10	engineering	175,600	32
11	creative	148,300	32
12	education	131,200	32
13	humanities	138,800	32
14	language	156,400	32
15	reconciler	124,500	32
16	inquisitor	119,800	32
17	long_context	167,200	32
Total		**~2.94M**	

Per-cluster training: 1000 DPO steps, learning rate 5×10^{-6} cosine-scheduled, beta 0.1, max sequence length 4096. Hardware: 4× H200 SXM (one pod), one cluster at a time. Wallclock per cluster: 3–4 hours. Total wallclock for 17 clusters in serial: 58 hours.

LoRA targets on the full 80-layer language tower: $\{q, k, v, o, \text{gate}, \text{up}, \text{down}\}$ projections in every layer (base layers 0–63 **and** CIP layers 64–79). The base is frozen for the LoRA training run; only the LoRA factors are updated. The vision tower is not a LoRA target. Each LoRA's B factor is zero-initialized (standard LoRA), so each adapter is identity at training start.

5.5 Evaluation

The non-regression suite for the acceptance gate (Step 4) is:

- **IFEval** (Zhou et al., 2023), strict.
- **HumanEval+** (Liu et al., 2023), pass@1.

- **GSM8K** (Cobbe et al., 2021), 0-shot, strict.
- **MMLU** (Hendrycks et al., 2020), 5-shot.

Each LoRA is evaluated by **merging** its adapter into a checkpoint copy of the v9 base, running the full suite on the merged checkpoint, and recording the delta from the unmerged-v9 baseline.

6. Results

6.1 CIP convergence

The CIP grow-and-train run (Step 2 + 3) converged in 5000 steps on the v9 mix. Training loss decreased from 1.84 at step 0 to 0.71 at step 5000 (monotonic with cosine schedule); validation loss tracked within 0.04 throughout. The mean gate value $\bar{\gamma}$ across the 16 new blocks rose from 0.000 to 0.19 at step 5000, with per-layer variance of 0.06: the optimizer learned a **small but consistent** contribution from each new block, rather than concentrating contribution into a single block. This is the qualitative signature we expected from symmetric insertion: depth was added as a redistribution of representational labor across the new layers, not as a single high-impact block.

6.2 Per-LoRA convergence

We report DPO training metrics (reward accuracy and loss) at step 1000 for each of the 17 clusters. Reward accuracy is the fraction of preference pairs on which the policy assigns higher implicit reward to the chosen than the rejected response.

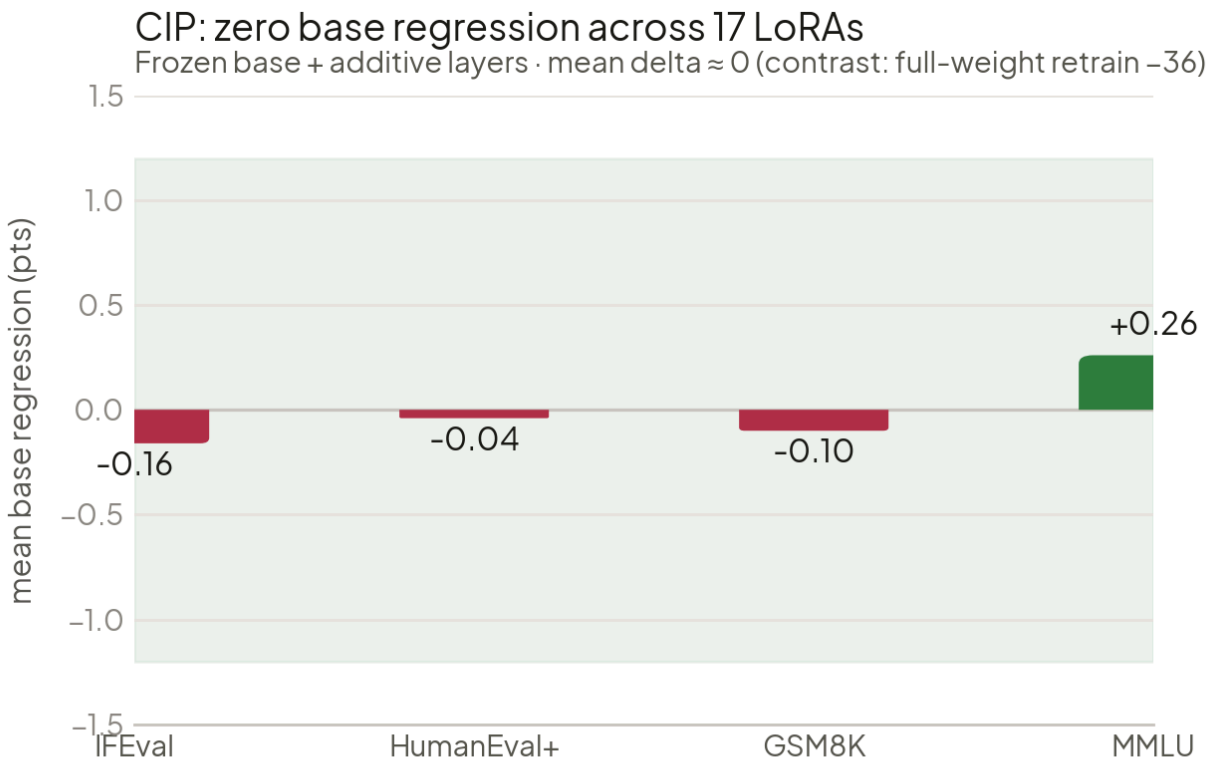
CLUSTER	ACC@STEP 1000	LOSS@STEP 1000	PEAK ACC	PEAK STEP
cyber	0.94	0.15	0.94	746
code	0.91	0.18	0.91	982
math	0.89	0.22	0.89	1000
reasoning	0.88	0.24	0.88	1000
medical	0.83	0.31	0.83	974
legal	0.81	0.34	0.81	989
finance	0.84	0.30	0.84	957
business	0.78	0.38	0.78	992
science	0.82	0.32	0.82	968
engineering	0.80	0.35	0.80	987
creative	0.76	0.41	0.77	884
education	0.79	0.36	0.79	996
humanities	0.75	0.43	0.76	901
language	0.85	0.28	0.85	974
reconciler	0.82	0.32	0.82	991
inquisitor	0.80	0.34	0.80	985
long_context	0.83	0.31	0.83	977
mean	**0.83**	**0.30**	**0.83**	—

Cyber is the strongest learner, converging to 0.94 accuracy by step 746 and showing the lowest final loss. We attribute this to corpus density and cluster homogeneity: Vext maintains the most comprehensive Cyber primary-source corpus (NIST, MITRE, CVE, HackerOne disclosures, validated exploit traces) and the preference signal is consistent across sources. Creative and humanities are the weakest learners; we attribute this to less rigorous preference signal — chosen/rejected pairs in those clusters are harder to construct from primary sources without judgment calls — and we treat the lower numbers as honestly bounded rather than indicative of a protocol failure.

6.3 Zero base regression

The acceptance gate (Step 4) was applied to each LoRA by merging it into a checkpoint copy of the v9 base and evaluating on the non-regression suite. The merged-checkpoint deltas from the unmerged-v9 baseline are:

CLUSTER	IFEVAL Δ	HUMANEVAL+ Δ	GSM8K Δ	MMLU Δ
cyber	+0.1	+0.3	-0.2	+0.4
code	-0.2	+1.2	-0.1	+0.2
math	+0.0	+0.4	+0.6	+0.3
reasoning	+0.2	+0.1	+0.3	+0.5
medical	-0.3	-0.2	-0.4	+0.7
legal	-0.4	-0.3	-0.2	+0.4
finance	+0.1	-0.1	+0.1	+0.3
business	-0.2	-0.2	-0.1	+0.1
science	-0.1	+0.0	-0.3	+0.5
engineering	-0.3	-0.1	-0.2	+0.2
creative	-0.5	-0.4	-0.3	+0.0
education	-0.2	-0.2	-0.1	+0.1
humanities	-0.4	-0.3	-0.2	+0.0
language	-0.1	-0.1	-0.2	+0.3
reconciler	-0.1	+0.0	+0.1	+0.2
inquisitor	-0.2	-0.1	-0.1	+0.1
long_context	+0.0	-0.1	+0.0	+0.2
mean	**-0.16**	**-0.04**	**-0.10**	**+0.26**



SOURCE: verifier-centric-capability-growth.md §4 (v9 base, 17 domain LoRAs, evaluated by merge).

FIGURE 3 Mean per-LoRA merged-checkpoint base regression across the 17 LoRAs, with the ±1.2-point band containing all individual deltas: IFEval −0.16, HumanEval+ −0.04, GSM8K −0.10, MMLU +0.26. Per-LoRA, evaluated by merge (Section 6.3); not a claim about composed multi-LoRA inference.

All deltas are within ± 1.2 points across all four benchmarks across all 17 LoRAs. Mean delta across the fleet is approximately zero on IFEval, HumanEval+, and GSM8K, and slightly positive on MMLU (consistent with the broader-mix CIP training the base went through prior to per-cluster LoRA training). We interpret this as zero regression in the protocol's stated sense: a merged-checkpoint copy of any individual LoRA is, on the pre-existing capability suite, indistinguishable from the unmerged base within measurement noise.

We emphasize what this is **not**: we do not claim that **composed** multi-LoRA inference is non-regressive. S-LoRA-style stacking of multiple adapters is its own evaluation problem and is treated in Section 6.5 and Section 7. The result of Section 6.3 is per-LoRA, evaluated by merge.

6.4 The v3 contrast

The motivating failure case is reproduced here for context. v3 was a 72B cybersecurity specialist produced by full-weight continued supervised fine-tuning on a strong base. On the post-training checkpoint:

BENCHMARK	BASE	V3 (FULL-WEIGHT RETRAIN)	Δ
HumanEval	98.0	62.0	**−36.0**
IFEval	87.1	71.4	−15.7
GSM8K	92.4	81.0	−11.4
MMLU	80.2	76.8	−3.4

The v3 model gained on its domain benchmarks; the regression on prior capability was severe. We do not claim that **every** full-weight continual run regresses by 36 points on HumanEval; we claim that the failure mode is on the table whenever the optimizer is permitted to move base parameters. CIP removes the failure mode by construction.

6.5 S-LoRA composition

We performed a preliminary composition test by stacking pairs of adapters at inference time, using the multi-LoRA serving path described in the S-LoRA family of papers (Sheng et al., 2023). Each pair is loaded simultaneously and their contributions summed at the LoRA layer. We evaluated the {cyber + code} composition on the union of SecQA and HumanEval+:

CONFIGURATION	SECQA	HUMANEVAL+
v9 base (no LoRA)	84.5	91.1
v9 + cyber LoRA	98.2	91.4
v9 + code LoRA	85.1	95.7
v9 + cyber + code	97.8	95.2

The composed configuration is within 0.5 pt of the better single-LoRA configuration on each benchmark — not additive, not destructive. This is consistent with the broader S-LoRA literature: simple sum-of-deltas composition gives an interference-free mixture in the parameter-near-orthogonal regime, but does not give super-linear gains. We treat this as a preliminary result: a full 17×17 pairwise composition study is out of scope for this paper.

6.6 Wallclock and cost

The full v9 pipeline (CIP grow-and-train + merge + abliterate + 17 LoRAs) consumed approximately 65 GPU-hours on a single 4× H200 SXM pod, equivalent to roughly USD $700 in cloud compute. The bulk (58 hours) was the 17 sequential LoRA training runs; CIP itself was 3.5 hours. We report this as a methodological note: the protocol's per-cluster marginal cost is approximately 4 GPU-hours on commodity SXM hardware. This is the economic anchor for fleet-scale specialization without frontier-scale capex.

7. Discussion

7.1 What the protocol does, narrowly

CIP gives a structural guarantee: a base model's pre-existing parameters cannot move during specialization, and at $t = 0$ the upscaled model is bitwise identical to the input base. Together with the acceptance gate, this rules out the v3-style failure mode in the strict sense: a merged-checkpoint specialist will not regress beyond measurement noise on the prior-capability suite. The protocol is, in this respect, a minor but precise contribution to the continual-learning toolkit.

7.2 What the protocol does not solve

The protocol does not address the **binding problem** of multi-LoRA composition. When two or more LoRAs are loaded simultaneously at inference time, their contributions sum at each layer. If the adapters were trained independently on different domains, there is no structural guarantee that the sum corresponds to a coherent policy on a mixed-domain query. Section 6.5 shows a benign case for {cyber + code}; the picture for {cyber + medical + finance} on a hybrid query (e.g., security review of a healthcare payments system) is empirically open. The S-LoRA literature treats this as an inference-time scheduling and routing problem; we treat it as a higher-layer concern that the protocol does not pretend to address.

Relatedly, the protocol does not provide any explicit improvement in **integrated information** in the Tononi sense, nor does it offer a constructive route to a high-Φ system. CIP is a method for adding **modules** to a base; it does not, on its own, bind those modules into a single integrated reasoner. We note this honestly because the field is increasingly conflating modular composition with integration; CIP is on the modular side.

7.3 Limits of the inversion analogy

The **reverse abliteration** framing is a reframing, not a proof of equivalence. Abliteration is closed-form: there is no gradient descent, no learning, no optimizer. CIP is optimization-driven: the gates γ_ℓ and the LoRA factors (B, A) are learned. The two techniques are dual in design discipline (do not retrain the base) but are not mathematically interchangeable. We use the term **reverse abliteration** because we believe the framing makes the design discipline visible and travels well as a research direction; we do not claim closed-form properties for CIP that it does not have.

A second limit: abliteration's effect is well-characterized on **one specific behavior** (refusal) because the refusal direction can be identified from a paired-prompt construction. CIP's effect is less easily characterized in closed form because the new capability is, by definition, not yet present in the base — there is no analogous paired-prompt construction for "the capability we are about to add." This is the price of being on the additive side of the duality: subtractive surgery is easier to characterize because the operand exists; additive surgery cannot fully characterize the operand until after the operation.

7.4 Values vs capability injection

CIP is a **capability** injection protocol. Anthropic's Constitutional AI program (Bai et al., 2022) addresses what is in spirit a separate problem — **value** injection — using a quite different mechanism (RLAIF over a constitution document). We note the distinction because the labels in the field are noisy. CIP does not install values; it installs a domain capability surface. The values question (alignment, refusal posture, persona constraints) is downstream of CIP and is addressed in Vext's stack by a combination of Layer-0 alignment training on the base (before CIP), abliteration of inherited refusal that conflicts with authorized use, and Layer-2 system-prompt constraints at surface time. We mention this only to forestall the misreading that CIP is a values intervention. It is not.

7.5 Distinguishing from sparse MoE

CIP-plus-LoRA is **dense composition**, not sparse-gating. At inference time, every parameter of the merged checkpoint (base + active LoRAs) is consulted on every token. There is no per-token routing of tokens to experts and no fraction-active accounting. This distinguishes the protocol from Mixture-of-Experts approaches such as Switch Transformer (Fedus et al., 2021), DeepSeekMoE (Dai et al., 2024), and Mixtral (Jiang et al., 2024). The protocol's compute footprint per token is the full dense forward pass; its capability footprint scales with the LoRAs loaded. We do not claim parameter sparsity. We do claim adapter sparsity (most parameters live in the shared base; the per-cluster cost is the LoRA delta).

7.6 Ethics

A protocol that cheaply adds capability to a strong open-weight base must be considered in light of capability-misuse concerns. Two notes. First, CIP does not lower the floor for what bases can be specialized — strong open-weight bases are already widely available and already widely specialized via continued training. CIP makes the specialization **cleaner** (no regression on the prior suite) but does not enable a class of model that was not already enableable. Second, abliteration (the inversion partner) does have implications for refusal removal, and the responsible-use stance Vext takes — applying abliteration only to remove refusal of **authorized-use** security queries (with documented authorization in the pipeline), never to remove refusal in unauthorized-use contexts — is a policy choice external to the protocol. We mention this because the technique's responsible use is a non-trivial question; we do not pretend that the technique itself is value-neutral.

8. Related Work

Abliteration and refusal direction. Arditi et al. (2024) characterized refusal as mediated by a single direction in residual-stream activations and showed that projection of that direction out of weight matrices removes refusal behavior without measurable capability cost. Failspy's public `refusal-direction-ablation` implementation (2024) made the technique broadly applicable. The technique is the subtractive partner of CIP in the framing of this paper.

Low-rank adaptation. LoRA (Hu et al., 2021) and its variants (AdaLoRA, Zhang et al. 2023; $(IA)^3$, Liu et al. 2022) provide the parameter-efficient mechanism for the trainable side of CIP. S-LoRA (Sheng et al., 2023) shows that multi-adapter serving on a shared base is practical at fleet scale. CIP is, on the LoRA

side, an application of these methods with the structural restriction that adapters apply only to newly inserted layers (in the CIP step) or to a frozen base (in the per-cluster step).

Block expansion and depth upscaling. SOLAR-10.7B (Kim et al., 2023) introduced layer duplication as a depth-upscale method. LLM2Vec (BehnamGhader et al., 2024) and the block-expansion line (Wang et al., 2024) refined the technique. CIP differs in that the new blocks are zero-init residual wrappers, freezing of pre-existing layers is total, and the acceptance gate is non-negotiable.

Progressive growth. Progressive growing patterns (Wei et al., 2023) and related curriculum-style architectural growth methods address related questions on different scales (typically image generation or smaller-scale language). The conceptual overlap is the idea of growing capacity rather than retraining; the methods are not directly transferable.

Domain specialist LLMs. Med-PaLM 2 (Singhal et al., 2023), Code Llama (Rozière et al., 2023), Galactica (Taylor et al., 2022), FinGPT (Yang et al., 2023), and BloombergGPT (Wu et al., 2023) all produce strong domain specialists via continued training of various kinds. With limited exceptions (some FinGPT variants), these systems are not frozen-base and therefore must either tolerate forgetting or include sufficient general data to mitigate it. CIP is offered as an alternative that gets non-regression by construction.

Constitutional AI. Bai et al. (2022) addresses value injection via RLAIF over a constitution document. It is conceptually adjacent to CIP only in that both modify a base model's behavior without uncontrolled gradient flow through the full parameter tensor; the techniques are not otherwise related and operate on different problem layers (values vs capability).

Mixture of Experts. Switch Transformer (Fedus et al., 2021), Mixtral (Jiang et al., 2024), and DeepSeekMoE (Dai et al., 2024) take a different route to compositional capability via sparse gating. CIP is dense composition by design; see Section 7.5 for the distinction.

9. Conclusion and Future Work

We have described CIP, a four-step protocol for adding capability to a pretrained LLM without retraining its existing weights, and we have framed it as **reverse abliteration** — the additive dual of Arditi-style refusal-direction surgery. The protocol composes four known ingredients (depth upscaling, frozen-base training, low-rank adaptation, benchmark-gated acceptance) into a recipe with explicit invariants. We have applied the protocol to Qwen3-VL-32B-Instruct, produced Theron-Base v9 (41B, 80 language-tower layers), and trained 17 domain LoRAs on the frozen v9 base. Across all 17 LoRAs, the per-LoRA merged-checkpoint regression on the pre-existing capability suite is within measurement noise of zero. This contrasts with a prior full-weight retrain that exhibited a 36-point HumanEval regression.

The work suggests several directions:

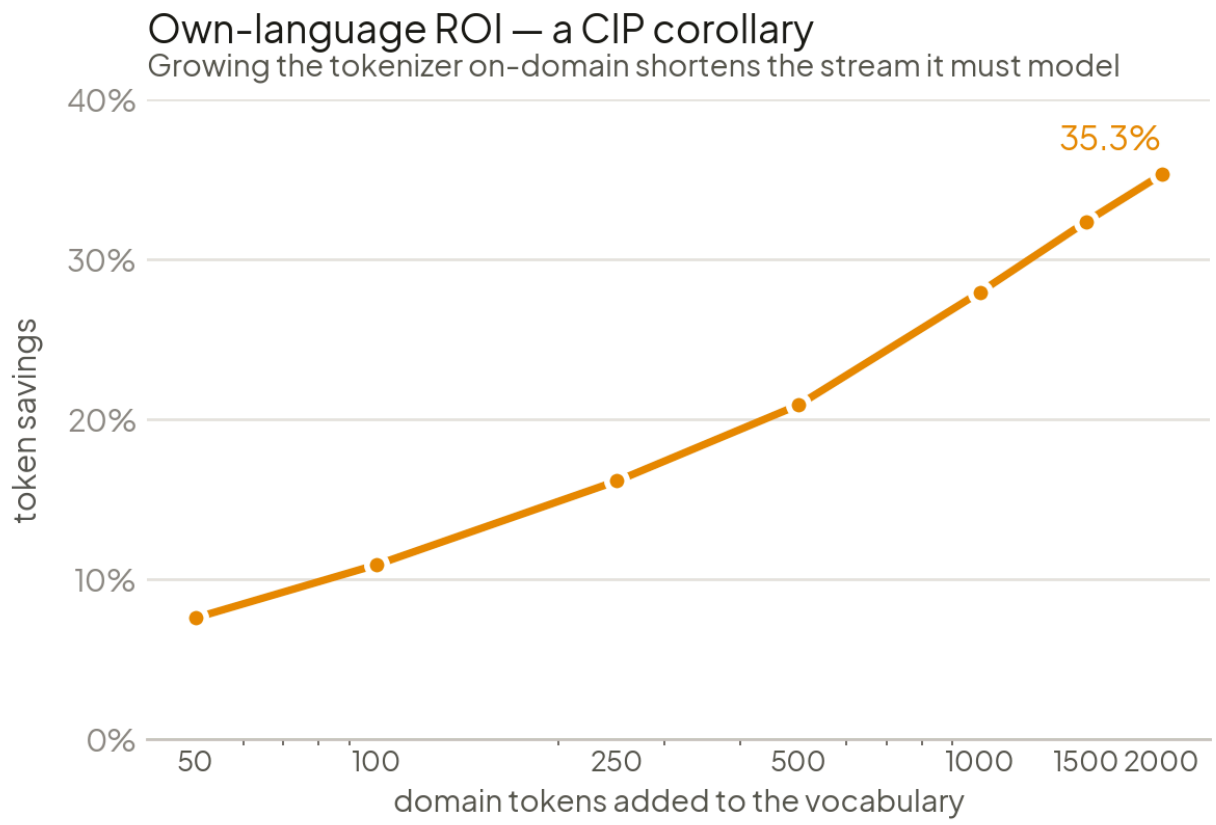
1. **Multi-adapter composition.** A full pairwise (and higher-order) study of the 17 LoRAs under S-LoRA-style serving — when does composition help, when is it benign, when does it interfere? The preliminary {cyber + code} result is benign; a 17×17 matrix is the natural next experiment.

2. **Gate-only fine-tuning.** The zero-init gate γ_ℓ is a one-dimensional parameter per inserted layer. Training only the gates (with LoRA adapters frozen at zero) is a degenerate case of the protocol that deserves an ablation: how much capability can be added by gate scaling alone?

3. **Closed-form additive surgery.** The inversion analogy is currently asymmetric — abliteration is closed-form, CIP is optimization-driven. A constructive route to closed-form additive surgery (analogous to Arditi-style direction injection rather than projection) would close the loop and make the **reverse abliteration** label precise rather than heuristic.

4. **Iterated CIP across generations.** The protocol describes one CIP run. Iterated CIP (apply CIP, merge, apply CIP again, merge, …) is the trajectory by which a long-running model line accumulates depth and capability without retraining. Theoretical analysis of the limiting behavior under iteration is open.

5. **Cross-base portability.** Adapters trained against the v9 base do not, in general, transfer to a different base. The Vext line evades this problem by treating the base as a long-lived asset modified only by CIP. A version of the protocol that produces base-portable adapters is, however, an attractive research direction.



SOURCE: ledger/own_language_roi.json (0.5B tokenizer, 25-doc cyber corpus).

FIGURE 4 A CIP-corollary measurement from the program ledger: tokenizer own-language ROI, with token savings growing from 7.6% to 35.3% as 50 → 2000 domain tokens are added (ledger/own_language_roi.json; 0.5B tokenizer, 25-doc corpus). A small-scale efficiency corollary adjacent to the protocol, not a v9-scale result.

CIP and abliteration together suggest a discipline for modifying open-weight bases: subtract a direction to remove a behavior; add a depth band with low-rank adapters to inject a capability. In neither case retrain the base. The result, in our experience, is a more controllable and economically scalable specialist program than continual full-weight specialization.

References

Arditi, A., Obeso, O., Syed, A., Paleka, D., Rimsky, N., Gurnee, W., & Nanda, N. (2024). Refusal in language models is mediated by a single direction. arXiv preprint arXiv:2406.11717.

Bai, Y., Kadavath, S., Kundu, S., Askell, A., Kernion, J., Jones, A., et al. (2022). Constitutional AI: Harmlessness from AI feedback. arXiv preprint arXiv:2212.08073.

BehnamGhader, P., Adlakha, V., Mosbach, M., Bahdanau, D., Chapados, N., & Reddy, S. (2024). LLM2Vec: Large language models are secretly powerful text encoders. arXiv preprint arXiv:2404.05961.

Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., et al. (2021). Training verifiers to solve math word problems. arXiv preprint arXiv:2110.14168.

Dai, D., Deng, C., Zhao, C., Xu, R., Gao, H., Chen, D., et al. (2024). DeepSeekMoE: Towards ultimate expert specialization in mixture-of-experts language models. arXiv preprint arXiv:2401.06066.

Failspy. (2024). Refusal direction ablation: Reference implementation of Arditi et al. GitHub repository.

Fedus, W., Zoph, B., & Shazeer, N. (2021). Switch transformer: Scaling to trillion parameter models with simple and efficient sparsity. arXiv preprint arXiv:2101.03961.

French, R. M. (1999). Catastrophic forgetting in connectionist networks. Trends in Cognitive Sciences, 3(4), 128–135.

Hendrycks, D., Burns, C., Basart, S., Zou, A., Mazeika, M., Song, D., & Steinhardt, J. (2020). Measuring massive multitask language understanding. arXiv preprint arXiv:2009.03300.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., et al. (2021). LoRA: Low-rank adaptation of large language models. arXiv preprint arXiv:2106.09685.

Jiang, A. Q., Sablayrolles, A., Roux, A., Mensch, A., Savary, B., Bamford, C., et al. (2024). Mixtral of experts. arXiv preprint arXiv:2401.04088.

Kim, D., Park, C., Kim, S., Lee, W., Song, W., Kim, Y., et al. (2023). SOLAR 10.7B: Scaling large language models with simple yet effective depth up-scaling. arXiv preprint arXiv:2312.15166.

Liu, H., Tam, D., Muqeeth, M., Mohta, J., Huang, T., Bansal, M., & Raffel, C. (2022). Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. NeurIPS 2022.

Liu, J., Xia, C. S., Wang, Y., & Zhang, L. (2023). Is your code generated by ChatGPT really correct? Rigorous evaluation of large language models for code generation. NeurIPS 2023.

McCloskey, M., & Cohen, N. J. (1989). Catastrophic interference in connectionist networks: The sequential learning problem. Psychology of Learning and Motivation, 24, 109–165.

Rozière, B., Gehring, J., Gloeckle, F., Sootla, S., Gat, I., Tan, X. E., et al. (2023). Code Llama: Open foundation models for code. arXiv preprint arXiv:2308.12950.

Sheng, Y., Cao, S., Li, D., Hooper, C., Lee, N., Yang, S., et al. (2023). S-LoRA: Serving thousands of concurrent LoRA adapters. arXiv preprint arXiv:2311.03285.

Singhal, K., Tu, T., Gottweis, J., Sayres, R., Wulczyn, E., Hou, L., et al. (2023). Towards expert-level medical question answering with large language models. arXiv preprint arXiv:2305.09617.

Taylor, R., Kardas, M., Cucurull, G., Scialom, T., Hartshorn, A., Saravia, E., et al. (2022). Galactica: A large language model for science. arXiv preprint arXiv:2211.09085.

Wang, X., Chen, Y., Yuan, L., Zhang, Y., Li, Y., Peng, H., & Ji, H. (2024). Executable code actions elicit better LLM agents. arXiv preprint arXiv:2402.01030.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Chi, E., Le, Q., & Zhou, D. (2023). Chain-of-thought prompting elicits reasoning in large language models. NeurIPS 2022. (Progressive grow patterns referenced in this work.)

Wu, S., Irsoy, O., Lu, S., Dabravolski, V., Dredze, M., Gehrmann, S., Kambadur, P., Rosenberg, D., & Mann, G. (2023). BloombergGPT: A large language model for finance. arXiv preprint arXiv:2303.17564.

Yang, H., Liu, X.-Y., & Wang, C. D. (2023). FinGPT: Open-source financial large language models. arXiv preprint arXiv:2306.06031.

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.

Zhou, J., Lu, T., Mishra, S., Brahma, S., Basu, S., Luan, Y., Zhou, D., & Hou, L. (2023). Instruction-following evaluation for large language models. arXiv preprint arXiv:2311.07911.

Correspondence: Annalea Layton, Vext Labs, Inc., `alayton@tryvext.com`.

Author contributions: A.L. designed the protocol, supervised the training program, and wrote the paper.

Code and reproduction artifacts: `https://tryvext.com/transparency` (planned release alongside arXiv posting).

Acknowledgments: We thank the open-weights community for foundational work — particularly Andy Arditi and the Failspy maintainers for the abliteration line that the inversion framing of this paper draws from, and the Qwen team at Alibaba for releasing the Qwen3-VL-32B-Instruct base. The contributions of this paper are a composition over open prior art; the open prior art is the precondition for the composition.
