LLM-Directed Model Stitching and Weight Editing

A case study connecting a frozen Go specialist to a language model

Trevor Martin - DataStreams Solutions | Coauthor: GPT-6 Astra (AI model)

6 November 2026 | Research preprint; not peer reviewed

Abstract

Can an LLM coding agent engineer connections between pretrained models and directly improve those connections through weight edits? We document a human-directed, LLM-executed case study in which the Codex agent implemented and iteratively revised a learned hidden-state round trip between Qwen3.5-4B and a frozen 9x9 Go specialist. The agent wrote the interfaces, training and causal diagnostic code, then authored reversible tensor transformations that changed the learned receiver without gradient updates. Ordinary board text enters Qwen; a learned sender reconstructs expert inputs from Qwen hidden states; the expert computes a spatial representation; and a learned receiver returns it to later Qwen layers for native generation. Initial factual correctness increased from 28/64 with the interface disabled to 38/64 enabled. Across subsequent interventions, board occupancy reconstruction increased from 48.81% to 78.20%, and queried-square attention accuracy increased from 6/256 to 90/256. Direct edits to 1,152 receiver weights reduced mean answer-token negative log likelihood by 6.91% or 2.57%. These are measured, task-specific improvements from an agent-led engineering process; the direct edits also reduced generated correctness from 54/96 to 52/96 or 53/96. Shuffled expert memory often preserved most output gains. The study uses 128 training boards, 32 development boards, and 64 repeatedly inspected confirmation boards; factual generation covers only 16 boards. The evidence demonstrates agent-implemented neural stitching and structured weight editing, with improvements in particular interface and probability metrics. It does not yet establish broad strategic Go transfer or a general ability to optimize arbitrary networks.

1. Introduction

Specialist models can contain useful domain computations that are expensive to reproduce with language-model training alone. Reusing such a model inside a language model requires both an input interface that expresses the language model's state in a form the specialist can use and an output interface that makes the resulting representations useful downstream. Merely attaching an expert's final policy output does not test these two learned interfaces.

We investigate this distinction in a small, auditable Go case study. The target architecture is text -> language-model hidden states -> learned sender -> frozen specialist -> learned receiver -> later language-model hidden states -> language-model output. We call the pair of learned interfaces a suture. The term describes an implementation; it does not imply that model stitching is new, that the expert can be removed at inference, or that expertise has been absorbed into the original language-model weights.

The higher-level subject is the LLM agent as a model-engineering participant. Under human direction, it did substantive work beyond choosing a training command: it constructed the hidden-state interfaces, diagnosed sender and receiver bottlenecks, designed controlled interventions, and directly rewrote selected connector weights. We document three kinds of progress separately: an operational learned connection between heterogeneous frozen models; improved reconstruction and addressing within that connection; and probability-loss improvements from agent-authored direct edits. This is evidence of what an LLM agent can implement and improve in a concrete case, rather than a controlled comparison of agents or a claim of autonomous discovery without human guidance.

The experimental record includes 29 local stages, scaffold baselines, learned-interface trials, numerical qualification, causal controls, and reversible edits. Some improvements do not translate into better generated answers, and some output gains survive wrong-expert controls. Preserving these distinctions makes the agent

contribution testable: the implemented mechanism, the improved metric, and the remaining failure can each be inspected independently. The scope is an exploratory model-engineering case study, not a competitive Go benchmark.

2. Related work and scope of novelty

Model stitching connects frozen network portions through a trainable interface. Bansal, Nakkiran, and Barak [1] study stitching as a way to compare representations and transfer them between models. Our round trip differs operationally from a single prefix-to-suffix stitch, but connecting pretrained networks is established prior art. We make no first-of-its-kind claim for neural model composition.

Flamingo [2] and BLIP-2 [3] connect pretrained visual representations to language models through learned components. They motivate the use of attention-based interfaces while studying different inputs, datasets, and capabilities. Branch-Train-Stitch [4] is closer to bidirectional hidden-state communication: it connects frozen language-model experts and a seed model using learned stitch layers. The present case instead reconstructs a convolutional Go expert's structured input from language-model hidden states and returns its spatial trunk representation. This is a difference in experimental setting, not an established novelty or superiority claim.

The base language-model family is Qwen3.5 [5]; the donor is the pinned Moka implementation [6]. Low-rank adaptation [7] is relevant to the surrounding decision-model infrastructure, but the Go suture is an attention-based module rather than a LoRA update. The later direct edits alter the learned receiver weights without gradient steps. They neither replace the suture training nor establish a general method for reading semantic skills directly from arbitrary model weights.

3. Models and learned architecture

The language-model configuration has hidden width 2,560. A reader is attached after decoder layer index 8 and a receiver to the MLP output at layer index 28, using zero-based indexing. The host checkpoint includes a previously trained decision adapter, a choice residual, and experimental native/choice weight patches. Importantly, native Go forwards execute inside pair.base(): decision adapters are disabled and the choice-only residual is bypassed. The active native route retains the existing layer-16 base-weight edit. Thus these native results are not evaluations of an untouched stock Qwen model, nor evidence that the recent decision SFT adapter caused the Go results. The loaded 49,152-exposure SFT state is relevant to separate packed-route retention checks. Exact checkpoint and patch identifiers are listed in the supplement.

The donor is millionco/moka commit 3b249ec094c3f0fe0eb9b96144e25d65149954a9. Its pinned quantized artifact contains 244,529 scalar values and occupies 260,048 bytes. The local PyTorch port dequantizes according to the artifact manifest and freezes the resulting weights. We use the final spatial trunk, an 81-by-48 representation, rather than copying policy logits into Qwen's output. No tree search is used. Upstream versions and advertised parameter counts can differ; the commit and artifact hash define the donor used here, and upstream playing-strength claims are not measurements of this system.

The sender has 81 learned square queries of width 64. It attends to layer-normalized early Qwen hidden states through learned key and value projections. A linear map predicts 12 planes per square: sigmoid outputs for the first 11 planes and tanh for the signed komi plane. These are soft predictions; they are not projected to a guaranteed legal board. Ground-truth features supervise training losses, but are not passed into the deployed sender API.

A_s = softmax(Q_s K_s(H_r)^T / sqrt(64) + prompt mask); F_hat = phi(W_p A_s V_s(H_r)); M = E_trunk(F_hat).

A learned scalar sigmoid gate is computed from the last prompt token's early hidden state. The receiver attends from later Qwen states to the expert memory. Each 48-dimensional normalized expert vector is concatenated with fixed 9-dimensional row and 9-dimensional column one-hot codes. These positional codes enumerate expert-memory locations; they are not oracle query-square inputs. Receiver width is 64, and its output projection

maps back to 2,560 dimensions. The gate scales the residual write; it does not skip expert computation or constitute a learned invocation scheduler.

A_r = softmax(Q_r(LN(H_w)) K_r([LN(M); P])^T / sqrt(64) + B); Delta = g W_o A_r W_v [LN(M); P].

The initial receiver has B=0. Later stages introduce an address module predicting nine row scores, nine column scores, and a sigmoid strength from the final prompt-token state. Their sums form a learned 81-square attention bias. Query-square labels supervise this module only during training. The outer relevance gate and this address-strength gate are distinct.

Prompt ownership masks exclude padding and teacher-forced answer suffixes from the sender. Expert information is written only at the last prompt position and subsequent output positions, avoiding a write from full-prompt information into earlier tokens. Per-request memory is reset; cached decoding retains only that request's prompt-derived expert memory. Qwen's own vocabulary head generates the final text. The fixed prompts request short factual sentences or a move coordinate; this is not unrestricted explanation generation.

Table 1. Data partitions used by the learned round trip (006 onward). One board supplies five Go task families. Arithmetic examples are separate.

Partition	Go boards	Go rows	Arithmetic	Total rows
Training	128	640	128	768
Development	32	160	32	192
Confirmation	64	320	64	384

4. Data, objectives, and evaluation

The board inventory comes from an existing local Go TRAIN dataset, not a newly collected benchmark. Preparation replays game histories and checks the resulting board and side to move. It selects one admissible position per game, excludes the 128 games previously used in policy experiment 002, and assigns 224 games to group-disjoint construction partitions by a fixed hash order. The initial factual generator requires at least 24 plies. The factual families are occupancy, liberties, legality under simple ko with suicide forbidden, and immediate captures. Labels are computed by rules. A fifth family asks for the frozen donor's highest-scoring legal move; this target measures donor imitation, not optimal play. Arithmetic examples provide a narrow non-Go control.

Board text includes the diagram and state context. The preparation code uses a parser to validate serialization and produce supervision. During the learned round trip, only ordinary tokenized text and a prompt mask enter the interface; no parser-derived board planes or query coordinates are provided to it. Early stages 003-005 do use such structured features and therefore belong to a different architecture. The 224 games were group-disjoint for initial suture fitting, but their development and confirmation outcomes were repeatedly inspected while designing successors. Unknown overlap with upstream model pretraining also prevents a global unseen-data claim.

The first learned trial 006 trains the sender, receiver, and relevance gate while keeping the language-model and donor weights fixed. AdamW uses learning rate 0.001, zero weight decay, gradient clipping at 1, and batches of four for three epochs: 576 updates and 2,304 repeated prompt exposures over 768 unique training rows. Qualification updates are restored before retained learning. Answer cross-entropy is averaged over answer tokens, including the end marker, then over examples. Binary-plane reconstruction uses a positive-target weight of four, plus komi mean-squared error; relevance uses binary cross-entropy.

L = L_answer + 0.2 L_features + 0.1 L_gate.

Subsequent stages isolate components. Trial 007 fits only the sender to cached reconstruction targets for 400 updates. Trial 008 adds training-token square-attention supervision for 400 updates. Trial 009 adapts the receiver for 576 updates, freezing the improved sender and gate. Trial 011 adds receiver query-square supervision for 576 updates. Trial 017 fits only the new spatial-address module using cached frozen states for 480 updates. Trial 019

fits receiver value/output projections with class-balanced answer supervision for 576 updates. Each stage has its own recorded plan and optimizer state; these are adaptive sequential experiments, not independent replications.

Teacher-forced evaluation covers all 384 confirmation rows, reporting mean per-example answer-token negative log likelihood (NLL) and exact argmax agreement with all target tokens. Greedy native generation uses at most 24 new tokens. The generated subset comprises the first 16 fixed rows per family: 64 factual questions, 16 move prompts, and 16 arithmetic prompts. The 80 Go prompts come from only 16 boards, so 64 factual answers are not 64 independent boards. Development generation analogously uses eight rows per family, 48 answers total. Exact sentence matching is case-insensitive after trimming; legal-move scoring is separate from matching the donor's single preferred move. No games are played to completion.

Controls include disabled interface, zero expert memory, wrong-board expert memory, and later diagnostic interventions with correct board planes or exact query squares. Wrong-board tests cyclically shift memory across requests; cohorts are checked to contain distinct boards. Zero memory does not remove the fixed positional channels and is not identical to disabling the entire receiver. Correct-board and exact-square interventions explicitly bypass learned interfaces for diagnosis and are never counted as autonomous capability. Off-route source replay and weight restoration are separate from active-route retention.

5. Results

5.1. Structured-input scaffolds establish limited baselines

The first frozen policy-branch graft 001 exactly reproduced the standalone donor but regressed on an immediate-capture task: 50/128 parent answers versus 32/128 graft answers. On the task-matched recorded-move panel 002, the branch instead matched 53/128 recorded engine moves versus 6/128 for the parent, with 2/128 for shuffled policy outputs and 8/128 after a stem lesion. This useful module-reuse result does not show that Qwen internally reconstructed the board or acquired a strategic skill. Both panels were task-specific and the latter came from an existing training inventory.

In structured-input representation trial 003, typed factual correctness increased from 115/256 disabled to 159/256 enabled, but shuffled memory scored 161/256; native generation was 10/32, 15/32, and 18/32 respectively. An explicitly query-addressed follow-up 004 scored 221/256 using donor features and 214/256 using matched raw engineered features; both generated 28/32 correct answers. Its reported paired interval for donor-minus-raw accuracy included zero. Trial 005 reproduced the scaffold outputs through an explicit text parser. These stages motivated a learned round trip, but none implements text -> Qwen hidden -> expert input.

5.2. A learned round trip improves outputs, with weak donor separation

Table 2. Actual factual generation on the repeatedly used 64-question subset (16 boards). Wrong memory is a board-shuffle control; entries are point counts, not independent replication results.

State	Correct /64	Wrong memory /64	Legal moves /16
Interface off	28	--	4
006 joint interface	38	36	8
007 sender calibration	36	33	9
008 sender alignment	34	37	10
009 receiver adaptation	37	37	10
011 receiver alignment	38	36	10
017 spatial address	38	36	10
019 value adaptation	38	37	9
022 full position erasure	36	36	10
023 positional mean removal	37	35	10

Trial 006 gained ten factual answers over the disabled interface on this subset. However, shuffled memory retained eight of that ten-answer aggregate gain, and normal and wrong-memory conditions both produced eight legal moves out of sixteen. This supports a working interface that changes outputs, but weakly identifies use of the correct board's expert computation. A generic donor-mediated representation, learned answer priors, or other shortcuts remain viable explanations. The Go gate was near one and arithmetic gate near zero on these narrow tasks; arithmetic generation remained 16/16.

The sender initially reconstructed occupancy at 48.81%, below an always-empty baseline of 67.94%. Input-only calibration 007 improved this to 62.13%. Square-attention alignment 008 reached 78.20%, but the improvement was uneven: on move-view squares, own-stone recall was 311/824 and opponent-stone recall 537/838. Reconstructed-input donor policy argmax agreed with true-input donor argmax on only 2/64 boards after 008. Better occupancy reconstruction did not imply preservation of the expert's policy or better Qwen factual answers; factual generation fell to 34/64 at008.

Receiver adaptation 009 recovered three factual answers, to 37/64, but wrong-board memory also scored 37/64. Capture and liberties outputs often matched dominant training labels: 103/128 training capture examples have no capture, and 102/128 liberties examples have more than two liberties. These imbalances make aggregate answer correctness an insufficient indicator of board-specific computation.

5.3. Addressing and value decoding are distinct bottlenecks

A diagnostic on 009 found queried-square top-1 attention on only 1/256 factual confirmation prompts. Supervision011 increased this to 6/256. The learned spatial-bias module017 increased it to 90/256, while factual generation remained 38/64 and legal moves 10/16. Attention location is a descriptive interface metric; the expert trunk mixes context across squares, so it is not itself a causal certificate of correct reasoning. The observed increase is nevertheless a substantial improvement in the trained addressing behavior.

Class-balanced value/output fitting 019 reduced training loss but worsened NLL in every Go family on confirmation. Factual generation stayed 38/64, with different individual successes and failures, and legal moves fell from 10/16 to 9/16. We retain both 017 and019, including their partial per-answer gains, rather than treating a falling training objective as evidence of broad improvement.

Table 3. Development-only oracle interface diagnosis 021 for frozen017. These interventions are diagnostic bypasses, not deployable learned behavior.

Condition	Forced exact /128	Generated /32
Normal learned interfaces	93	24
Correct board planes	89	24
Exact query square	86	21
Correct board + exact query	82	19
Wrong board + exact query	79	22

Supplying correct board planes and exact query addressing did not broadly repair outputs. For017, exact-query intervention improved legality teacher-forced correctness from 17/32 to 24/32, but generated legality remained 5/8; other families worsened. This argues against treating perfect input reconstruction or addressing as a guaranteed downstream upper bound. Receiver coadaptation and distribution shift are plausible explanations, but the experiment does not isolate them as the sole cause.

5.4. Direct receiver edits expose probability-accuracy tradeoffs

These interventions were proposed and implemented by the AI coding assistant after inspecting the receiver architecture, saved outputs, and diagnostic controls. The assistant wrote executable tensor transformations, applied them to copies of the trained connector, and evaluated the resulting full composite model on the GPU. In particular, it edited writer.value.weight[:, 48:66], the 64-by-18 positional value block, rather than asking an optimizer to discover the changes. Thus the assistant participated directly in modifying the neural connection, beyond describing an architecture or drafting analysis.

Trial 022 evaluated six actual direct weight edits and controls from 017, restoring the same parent before every arm. Candidates doubled learned address scores, erased positional value columns, or combined the two; controls reversed these changes or used a norm-matched random address perturbation. Frozen development selection chose full positional erasure. Trial 023 then separated each 64-by-9 row/column value block into its repeated column mean and zero-mean contrast. Its two candidates removed one component or the other; controls added the components or subtracted a seeded norm-matched random constant direction. The prior 022 erasure was replayed exactly.

For each positional block P: M = (P 1 / 9) 1^T; C = P - M. The two candidates retain C or M.

With normalized attention, the repeated mean contributes the same value vector at every board square before output projection and relevance gating. The contrast retains location variation. This algebra motivates the intervention but does not label either component intrinsically useful or erroneous. Each edit changes 1,152 scalar values; no gradients, target fitting, or oracle inference inputs are used. Selection is by generated development correctness, then lower mean NLL, then fixed declared order. Only the two named candidates can advance; controls cannot. Selection is frozen before reused confirmation characterization.

Table 4. Direct edits on the same reused confirmation panel. NLL is mean answer-token loss on 384 rows; generation uses 96 prompts, including 16 arithmetic prompts. Lower NLL is better.

State	Mean NLL	Generated /96	Occupancy /16	Legal /16
017 reference	0.419906	54	8	10
022 full erasure	0.390898	52	6	10
023 mean removal	0.409095	53	7	10
022 wrong memory	0.389977	52	7	5
023 wrong memory	0.404200	51	7	7

Full erasure reduced NLL by 6.91% but lost two occupancy answers. Mean removal reduced NLL by 2.57%, restored both of those answers, and introduced a different black-to-white error; relative to 017 it fixed no incorrect generated answer and lost one correct answer. Legal moves remained 10/16, falling to 5/16 or 7/16 with shuffled expert memory. This is limited evidence that correct expert memory matters for some move outputs, not a playing-strength result. Wrong-memory aggregate NLL was slightly better than corresponding normal-memory NLL, further cautioning against reading pooled loss as a pure measure of correct expert use.

All seven 023 development states generated 32/48 correct answers. Mean removal improved NLL more than the fixed random-mean control, but that control also improved NLL; reverse component additions worsened it. Reverse/random arms were not all evaluated on confirmation, so no confirmation-wide specificity claim is made. The patches are retained as experimental tradeoff variants. Neither is adopted as a generally better model.

5.5. Protected projections and paired-board training

Trial 024 fitted a rank-two color span using 63 training boards and checked it on 65 other boards from the same training inventory. A direct edit removed the constant positional contribution orthogonal to that span, changing 1,152 weights without gradient updates. Check-set occupancy NLL improved from 0.1701690 to 0.1683686 (1.058%), while teacher-forced exact answers fell from 39/65 to 38/65. All six normal development conditions generated 32/48 correct answers. Geometric protection of a fitted span therefore did not establish behavioral preservation. Reverse, random, shrink and full-mean controls were retained; 017 remained the reference.

Passive capture 025 and CPU recovery 026 examined the learned path on 128 training boards. Queried-square attention was correct on 83/128, sender queried-cell occupancy on 90/128, and both on only 58/128. Own stones were reconstructed as opponent stones on 23 of 42 queried own-stone examples. A simple linear probe became less accurate after the value/output map, but the map had full column rank and an approximate inverse recovered the original value-probe predictions. Reduced linear-probe performance was therefore not evidence that the map destroyed color information. These are reused-training diagnostics, not autonomous task gains.

Trial 027 changed only receiver value and output projections from 017. It used 60 same-square, different-label training pairs plus replay, optimizing native answer CE and a paired log-odds hinge with weight 0.25 and margin 2. Across three epochs it retained 180 updates and 720 repeated prompt exposures over 189 unique rows; four qualification updates were discarded. Training took 84.89 seconds (8.48 exposures/s). Mean hinge fell from 1.6354 to 0.33125 and CE from 0.25124 to 0.17115. This combined-objective experiment does not isolate the hinge as the cause of any outcome.

Table 5. Actual generated answers for 017 and paired-board receiver 027 on the reused panel. Legal moves and preferred-move matches are different metrics.

Measure	017	027
Occupancy /16	8	9
Capture /16	11	7
Legality /16	9	9
Liberties /16	10	10
Preferred expert move /16	0	1
Legal moves /16	10	3
Arithmetic /16	16	16
All generated answers /96	54	52

The occupancy gain in 027 comprised five fixes and four regressions. Its outputs were nine white and seven black answers, with no empty answer; disabled and wrong-expert controls also achieved 9/16 occupancy. Thus this local gain did not demonstrate increased use of correct expert content. Legality and liberties NLL improved, but

capture, move and occupancy NLL worsened. One preferred-move match was gained while seven legal moves were lost. We preserve 027 and its partial gains, but retain 017 as the reference rather than describing 027 as an overall improvement.

5.6. CPU follow-ups limit the next hypotheses

CPU audit 028 tested whether player-relative own/opponent encoding explained the sender weakness. Adding player interactions did not improve any fixed raw color probe: sender check accuracy fell from 48/65 to 45/65 and expert-query accuracy from 39/65 to 35/65. A separate affine sender calibration 029 reduced occupancy BCE from 0.3371234 to 0.2849453 on the reused 65-board check, but correct cells fell from 4,317 to 4,102 out of 5,265. The reconstructed-logit approximation also required a separate runtime check. The patch was retained as a CPU tradeoff result and not rolled out on the full GPU model. Neither follow-up established a deployable fix.

6. Numerical controls, throughput, and failed experiments

All substantive model trials used finite guarded launches with exact process ownership, source hashes, checkpoint restoration, and terminal evidence. These operational checks make a run auditable; they do not prove scientific generalization. Direct-edit023 independently verified 4,461 pinned dependencies and 32 recorded artifacts, reproduced the 017 baseline and022 erasure, and checked exact undo and off-route source restoration. Such exact checks apply to fixed implementations and batch cohorts, not universal bitwise equivalence across hardware or batch shapes.

An attempted B4 acceleration012 failed its combined numerical/speed assertion before retained updates. Follow-up013 persisted the measurements: B4 was about 1.82 times faster but its mean NLL difference of -0.030355 exceeded the pre-existing absolute 0.03 bound. Component localization014 identified substantial amplification in the sender/expert path. Autonomous singleton sender extraction followed by batched full forward passed the existing bounded checks in 015/016, with measured speedups 1.263 and 1.400 on 8 and 48 reused inputs. Individual differences as large as 0.1221 NLL remained; passing a mean bound is not exact per-example numerical equivalence. Cached generation continued to use its original path, and full-model backward through the new wrapper was not qualified.

Other failures were implementation errors:015 completed numerical work but failed final verification because a variable holding source parameters was shadowed;018 shadowed an answer-class weight mapping before its first backward step;020 accidentally used B8 for a reference saved under B4 and failed replay. Corrected versions preserved the failed evidence and restored the intended cohorts rather than widening tolerances. The initial preparation also failed by importing an Apple MLX initializer on Linux; the repair used audited pure board/feature modules. None of these incidents is reported as a retained learning improvement.

The full model was loaded once per bounded comparison. Representative measured rates were 11.82 prompt exposures/s during006 joint-interface updates, 8.51 during019 value/output updates, and 8.93 questions/s for 023 warmed teacher-forced evaluation. The007 cached sender fit achieved 5,363 repeated exposures/s only after 48.62 seconds of activation extraction; this is not full-model training throughput. Recorded generation timings concern short answers, not long reasoning tokens/s. Runs used an RTX 3090 at 225 W with 85 C and 21 GiB supervision limits;023 peak reserved memory was about 10.09 GiB. We do not infer energy savings, steady-state optimal batching, or equivalence to an unrelated 27B serving benchmark.

7. Limitations and required next tests

The central limitation is adaptive reuse. Later mechanisms and edits were chosen after observing earlier development and confirmation outcomes. Stage-local selection was frozen before each confirmation, but this does not restore study-wide independence. We therefore report descriptive counts and paired changes without new significance declarations. Questions within a board are correlated, generation uses only 16 boards, there is one recorded initialization lineage rather than multiple independent seeds, and the corpus is not guaranteed absent from donor or language-model pretraining.

The current factual prompts use a single structured textual style and explicit allowed answer sentences. This constrains language output but does not supply the correct answer. The study does not demonstrate arbitrary-prose board understanding, faithful strategic explanations, self-inspection of reasoning, board-image input, packed decision-output compatibility for the learned round trip, complete-game play, Elo gains, or donor-free internalization. The expert remains part of inference. Learned gating was tested largely against arithmetic, not a broad set of tasks.

Off-route exact replay shows that disabling the branch restores selected original outputs. It does not show that the active branch preserves broad language ability. The frozen native host also contains a prior experimental weight edit; a stock-Qwen replication is missing. The matched raw-feature baseline in 004 uses an explicit scaffold and does not substitute for a learned sender/receiver baseline without a pretrained expert. A capacity-matched randomly initialized expert, a learned identity/raw-feature bottleneck, and trainable versus frozen expert comparisons are necessary to isolate the value of pretrained Go computation.

A stronger study would freeze one complete architecture and evaluate fresh game-disjoint, class-balanced boards with multiple seeds; measure paired results clustered by game; test rollout stability and complete games against fixed opponents; and include counterfactual board edits, wrong-square/wrong-board controls, and explicit legality repair policies. Receiver changes should be fitted only on construction data and tested without tailoring to the inspected errors. Broad retention and image-input experiments require separate data and qualification. These are proposed tests, not completed results.

8. Conclusion

This case shows an LLM coding agent carrying out model stitching and direct weight editing under human direction. It implemented a learned route from language-model hidden states into a frozen specialist and back, improved the sending and receiving interfaces through targeted experiments, and wrote structured receiver-weight changes that measurably lowered answer-token loss without gradient updates. The agent was an active implementer and experimenter, not merely a narrator of externally produced results.

The demonstrated improvements are specific: reconstruction and address accuracy improved, initial factual outputs improved over a disabled branch, and direct edits improved pooled probability loss while losing some generated answers. Wrong-expert controls and repeated small panels constrain the interpretation. The next scientific question is whether this engineering workflow yields reliable specialist transfer and jointly better outputs on fresh tasks, across seeds and experts. The current evidence supplies an implemented example, explicit mechanisms, useful gains, and preserved counterexamples from which to test that question.

9. Reproducibility, availability, and disclosure

Public release: GitHub https://github.com/tcmartin/qwen-go-suture and Hugging Face https://huggingface.co/trevdatastreams/qwen3.5-4b-go-suture. These repositories provide the research implementation, model artifacts and release-specific reproduction instructions. This paper is a research preprint, not peer reviewed. The reference release is stage 017; direct edits and later checkpoints are labeled experimental tradeoff variants, not promoted as universally improved models.

The minimal native-generation package uses the frozen Qwen base, the active native layer-16 patch, the frozen Go expert and learned suture. The historical native pair.base() route disabled decision adapters and bypassed the choice-only residual. Consequently, the 49,152-exposure decision SFT checkpoint is a historical loaded dependency for packed-route retention, not a required source of the native Go behavior reported here. Reproduction should follow the released loader and artifact hashes rather than replaying private absolute paths. Native-generation portability does not imply qualification of packed decision outputs or full-game play. The public stage-017 package includes the native layer-16 patch and donor weights, with deterministic scripts for the 022 full-erasure and 023 mean-removal edits. Its extracted portable native loader has passed exact CPU component and state checks. Full-model GPU replay of that extracted loader has not yet been performed; the GPU results in this paper come from the original research runtime, not a new public-loader evaluation. The

resolved upstream Qwen revision is 851bf6e806efd8d0a36b00ddf55e13ccb7b8cd0a; the public manifest supplies artifact verification details.

9.1. Human direction and AI-assisted research contributions

Trevor specified the specialist-to-general-model research goal, directed the change from an external parser/expert scaffold to a hidden-to-expert-to-hidden architecture, requested causal controls and direct weight interventions, and authorized the experiments. The Codex AI coding assistant translated those requests into implementation and experimental plans, wrote and revised the sender/receiver and diagnostic code, ran bounded GPU experiments through the guarded launch infrastructure, examined saved activations and outputs, and prepared the analysis and this manuscript. These are substantive implementation and experimental contributions, not only copyediting assistance. At Trevor Martin's request, GPT-6 Astra is credited as an AI coauthor; it is not a human author or an independent accountable legal person. Trevor Martin is the human contact for this release.

The actual stitching combined gradient-trained interfaces with assistant-authored direct weight edits. The learned sender, relevance gate, receiver, and spatial address module were fitted using the objectives described above. In stages 022 and023, the assistant then directly rewrote selected learned receiver tensors through code, without further gradient updates: it scaled address parameters, erased positional values, and decomposed positional weights into constant and contrast components for reversible removal/addition tests. The mean-removal edit changed 1,152 scalar weights and reduced reused-panel NLL by 2.57%; full erasure reduced NLL by 6.91%. Their one- and two-answer generation losses are reported alongside those gains. No assistant-generated answers were used as supervision for these direct-edit trials; their targets came from the existing rules/donor curriculum.

This is a hybrid human-directed, AI-executed model-engineering workflow. The direct edits act on a trained connector, not on the frozen Go donor or the original Qwen backbone in these stages. They show that an assistant can inspect a specific interface and implement testable, structured weight changes, but do not establish that it can infer an arbitrary skill from raw weights or independently invent the underlying specialist computation. All proposed edits, controls, failed runs, exact undo checks, and adverse outcomes remain part of the record. The numerical reductions are measured results; the causal interpretations remain hypotheses constrained by the controls.

Trevor Martin - DataStreams Solutions is the named human author; GPT-6 Astra is credited as AI coauthor for the implementation, experiment design, direct tensor edits and analysis described above. Publication is authorized by Trevor Martin. This is a public research preprint, not a peer-reviewed acceptance. Base models and upstream components retain their respective licenses; the release repositories identify included files and dependencies. The paper does not claim access to hidden training data or semantic knowledge identifiable from weights alone.

Appendix A. Complete experiment chronology

Table A1. All numbered Go stages through 029, including failed runs. Stage identifiers are local provenance labels.

Stage	Purpose and outcome
001	Frozen output graft; capture 50/128 ->32/128; donor port parity passed.
002	Task-matched policy branch:6/128 ->53/128 recorded moves; structured inputs.
003	Structured donor representations: native 10/32 ->15/32; wrong memory18/32.
004	Explicit square addressing: donor 221/256 vs raw 214/256; both native 28/32.
005	Text parser scaffold:96 format cases reproduced outputs; zero updates.
006	First learned hidden round trip; 576 updates; factual 28/64 ->38/64.
007	Sender reconstruction calibration; 400 cached updates; occupancy 62.13%.
008	Sender square attention; 400 updates; occupancy 78.20%, factual 34/64.
009	Receiver adaptation; 576 updates; factual 37/64, same as wrong memory.
010	Frozen diagnosis: receiver square top1=1/256; donor policy mismatch.
011	Receiver address supervision; 576 updates; top1=6/256; factual 38/64.
012	Spatial-readout attempt; qualification failure; zero retained updates.
013	B1/B4 diagnosis; faster B4 failed existing numerical bound.
014	Own-input component transplants localized batching drift; zero updates.
015	Singleton-sender qualification; measurements saved, final verifier bug.
016	Recovered verification and 8-input exact replay; zero updates.
017	Cached spatial readout; 480 updates; address 90/256, factual 38/64.
018	Value-readout attempt; class-weight variable shadowing before backward.
019	Corrected value fit; 576 updates; all Go NLL worsened, factual 38/64.
020	Oracle diagnosis attempt; accidental B8 reference mismatch; zero updates.
021	Corrected B4 factorial diagnosis; perfect inputs did not broadly repair.
022	Six direct edits/controls; erasure NLL-6.91%, generation54->52/96.
023	Six component edits/controls; mean removal NLL-2.57%,54->53/96.
024	Protected color projection: check NLL -1.058%, exact 39->38/65; zero gradients.
025	Passive path capture finished; final storage assertion failed; artifacts preserved.
026	CPU-only manifest/review recovery of 025; no repeated GPU capture.
027	Paired receiver:180 updates; occupancy8->9/16, legal moves10->3/16.
028	CPU perspective audit: no fixed raw probe improvement.
029	CPU affine sender patch: BCE improves, check cells4317->4102/5265; no GPU rollout.

Appendix B. Reproduction details and artifact identities

The principal files are go_hidden_roundtrip.py (sender, gate, ownership), go_native_bridge.py (frozen expert trunk and receiver), go_spatial_readout.py (address bias), go_receiver_sender.py (qualified singleton sender wrapper), go_position_components.py (023 transformations), and their versioned worker/reviewer/launcher modules. The historical audit retains absolute paths, stage configuration hashes, seeds, source checkpoints and pins; the public release uses a portable manifest and selected evidence records. Copying only a connector checkpoint without its native base edit and route settings does not reproduce this system.

Table B1. Key identities (full SHA-256 values are in provenance.json).

Artifact	Identity
Donor commit	`3b249ec094c3f0fe0eb9b96144e25d65149954a9`
Donor quantized weights SHA	`a7718c630cc52945d900f32db150dc633d88acb0af84e82069a545ebd8057ca9`
Suture curriculum SHA	`b5578bb5526c9a73f1d4170df2e614b7605b8272e658c9924c0139d3ab5844c2`
Loaded SFT 49152 state SHA	`95d904d0502a60b68a8fca007860a6ef7a127444cf0ee99fa9dca5e3a901cf95`
Active native layer 16 matrix SHA	`0964c602dbbacff506cb52761d3c771b7118560160243f8a7d0096cb7fc6cfd0`

For each candidate, restore the full 017 bridge; apply exactly one declared edit; save original and edited tensors; run unchanged fixed B4 cohorts; evaluate actual generation with the original cached path; freeze development selection; evaluate confirmation and wrong-memory control; restore and compare all source tensors. The023 random constant control uses CPU seed 2026110523 and matches each positional block's mean-vector norm independently. The candidate order is remove_mean then remove_contrast. Independent review reconstructs each patch from the parent and rejects altered patch contents even if a file-level manifest is recomputed.

The historical artifact-analysis script is CPU-only and does not claim to rerun all model evaluations. It reconstructed the reported 022/023 metrics from raw outputs; publication adds selected independently reviewed 024-029 records. Earlier numerical claims are cross-referenced to retained independent reviews and raw-artifact manifests. A portable reproduction should additionally verify all checkpoint hashes, upstream model revision, dependency versions, and exact runtime numerical behavior before any launch. No new evaluation was performed while drafting this paper.

References

[1] Bansal, Y., Nakkiran, P., and Barak, B. (2021). Revisiting Model Stitching to Compare Neural Representations. NeurIPS 34. https://arxiv.org/abs/2106.07682

[2] Alayrac, J.-B., Donahue, J., Luc, P., et al. (2022). Flamingo: a Visual Language Model for Few-Shot Learning. NeurIPS 35. https://arxiv.org/abs/2204.14198

[3] Li, J., Li, D., Savarese, S., and Hoi, S. (2023). BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models. ICML, PMLR 202:19730-19742. https://proceedings.mlr.press/v202/li23q.html

[4] Zhang, Q., Bhargava, P., Bi, C., Cai, C. X., Foerster, J. N., Fu, J., Koura, P. S., Silva, R., Shen, S., Dinan, E., Gururangan, S., and Lewis, M. (2025). BTS: Harmonizing Specialized Experts into a Generalist LLM. EMNLP, 6816-6834. DOI:10.18653/v1/2025.emnlp-main.347. https://aclanthology.org/2025.emnlp-main.347/

[5] Qwen Team (2026). Qwen3.5-4B model card. Accessed 5 November 2026; local model and route hashes determine this study. https://huggingface.co/Qwen/Qwen3.5-4B

[6] Million Software (2026). Moka: a Go-playing model. Pinned source commit 3b249ec094c3f0fe0eb9b96144e25d65149954a9, MIT license. Version-specific artifact verified locally. https://github.com/millionco/moka

[7] Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. (2021). LoRA: Low-Rank Adaptation of Large Language Models. arXiv:2106.09685. https://arxiv.org/abs/2106.09685

[8] DataStreams Solutions (2026). Public research code and reproduction instructions. https://github.com/tcmartin/qwen-go-suture

[9] DataStreams Solutions (2026). Qwen Go suture research model artifacts. https://huggingface.co/trevdatastreams/qwen3.5-4b-go-suture