Is 4-bit the Ceiling? Fitting North-Mini-Code into 24 GB on Apple Silicon, and a Streaming GPTQ for Large MoEs

Tony Salomone
Transformer Lab
tony.salomone@lab.cloud

Ali Asaria
Transformer Lab

Deep Gandhi
Transformer Lab

Abstract

We ask whether a 30B-total / 3B-active mixture-of-experts (MoE) coding model, *North-Mini-Code-1.0* (128 experts, top-8; Apache-2.0), can be made *better* than its vendor 4-bit quantization while still fitting the hard ≤24 GB resident budget of a 24 GB-class Apple Silicon machine. Our answer to the title's question is *yes* — 4-bit is the ceiling for this model within what we tested — which, against our goal of beating it, is a careful negative result.

Across a structured search — uniform bit-width, group size, number format, built-in mixed-bit predicates, a North-aware custom per-expert allocation, and a calibrated streaming GPTQ — no route that fits the budget beats round-to-nearest (RTN) 4-bit on coding evaluation, including configurations that spend *more* memory than 4-bit (not just cheaper ones). The two most direct threats both tie RTN on full HumanEval-164: the in-budget higher-bit `mixed_4_8` (5.2 average bits, 20.5 GB) scores 0.9024 vs. RTN's 0.8902 (148 vs. 146 of 164; McNemar exact $p = 0.79$, a +1.2% gap below our 1.5% margin), and calibrated streaming-GPTQ scores 0.8841 ($p = 1.000$). A cheap MoE→dense distillation pilot also fails its triage gate (0/32).

Two findings share the headline. First, no route beats 4-bit at ≤24 GB. Second, under our single-shot scaffold the practical limiter on agentic coding is not bit-width but *generation-length instability* (repetition loops): the 4-bit model fits in 20.81 GB at a real 32K-token agentic turn and sustains 40.2 tok/s, yet 21/30 SWE-Bench Verified instances looped to the 9000-token cap, 20 of them emitting no patch.

Our second technical contribution is the instrument that makes the first credible: a memory-efficient *streaming* GPTQ for large MoEs that quantizes a 60 GB / 128-expert model on a 48 GB Mac at 3.75 GB peak — where stock mlx-lm GPTQ is infeasible (whole-model plus all-layer Hessians resident). The primary success criterion (beat the published 4-bit quant) was **not met**.

1 Introduction

A growing class of developers treats a 24 GB-class Apple Silicon machine (e.g. a Mac mini) as a dedicated, private, always-on inference box for agentic coding. The constraint that defines this setting is a hard one: the model's weights, its KV cache for a realistic agentic turn, and activation overhead must all fit within roughly 24 GB of unified memory, while decoding fast enough (∼20 tok/s) to be usable. The natural candidate is a strong, openly-licensed coding model that already ships in many quantizations. We study *CohereLabs/North-Mini-Code-1.0* [1] — a 30B-total / 3B-active

MoE (`Cohere2MoeForCausalLM`; 49 layers; 128 experts, 8 active; no shared experts; a dense prefix layer; interleaved 1:3 global/sliding-window attention; hidden size 2048; expert intermediate 768; vocabulary 262144; bf16), trained by SFT then RLVR for agentic coding and released under Apache-2.0.

The deliverable was framed as "the single best coding model that runs in ≤24 GB on Apple Silicon (via MLX)," with an explicit bar: *beat the vendor's published 4-bit quantization on coding evaluation*, measured under our own harness, at ≤24 GB and ≥∼20 tok/s. We treated compression and a gated MoE→dense distillation as competing *means* to that end. This paper reports the outcome: that bar was not cleared, and the reason is informative. No route that fits the ≤24 GB budget — cheaper *or* higher-bit — beats North's 4-bit quantization on coding eval; across uniform, mixed-bit (including the in-budget higher-bit `mixed_4_8`), North-aware, and calibrated-GPTQ routes, every candidate ties or loses to it — so within our search there is no accuracy to recover inside the budget. The binding practical limitation for *agentic* use turns out not to be the memory budget — which is comfortably met — but the stability of long-form generation.

Making this finding credible required an instrument. The strongest accuracy-recovery routes (calibrated GPTQ/DWQ) are, as shipped in mlx-lm, infeasible on a 128-expert MoE on the available 48 GB development machine, because they hold the full bf16 model and all-layer Hessians resident, and the 128-expert `SwitchLinear` Hessians blow up to hundreds of GB. We therefore built a memory-efficient *streaming* GPTQ for large MoEs; it is independently useful and is the tool that lets us test the strongest compression hypothesis at all.

Contributions.

1. **An empirical finding: within what we tested, 4-bit is the ≤24 GB ceiling for North-Mini-Code** (§5). No in-budget quantization route we tried — uniform, group-size, number format, built-in mixed-bit (including the higher-bit `mixed_4_8`), North-aware custom per-expert allocation, or calibrated streaming-GPTQ — and no cheap MoE→dense distillation beats RTN 4-bit on coding eval; the two closest configs (`mixed_4_8` and streaming-GPTQ) are statistically indistinguishable on full HumanEval-164 ($p = 0.79$, $p = 1.000$) and within our equivalence margin (§5).

2. **A characterization of the agentic limiter: generation-length instability, not bit-width** (§5, §6). Under our single-shot scaffold the dominant failure is non-termination / repetition looping rather than wrong logic; memory and throughput clear their budgets comfortably, so the highest-leverage next step is decoding and stop-criteria tuning, not bit allocation.

3. **A memory-efficient streaming GPTQ for large MoEs** (§3). It quantizes a 60 GB / 128-expert model on a 48 GB Mac at ∼3.75 GB peak by processing one layer at a time and accumulating per-expert Hessians over routed tokens only — where stock mlx-lm GPTQ is infeasible. It is independently valuable and is the instrument that makes the negative finding credible (§5).

We deliberately do *not* claim our port of the `cohere2_moe` architecture into mlx-lm as a contribution: a concurrent upstream effort exists [2], and we mention our port only as an independent enabler used for this study.

2 Related Work

4-bit is the operating point; sub-4-bit craters code. The literature is unusually decisive about where to anchor. On Qwen3-30B-A3B [3] — a near-twin of North in shape (30B-total / 3B-active, 128 experts, top-8) but not identical (North has no shared experts, a dense prefix layer, and interleaved 1:3 global/sliding-window attention) — Lasby et al. [4] show that 4-bit weight-only quantization beats every pruning/distillation alternative at equal-or-smaller size, and that dropping below 4-bit collapses coding accuracy (W2A16 EvalPlus 28.6 vs. 77.6 at W4A16). The same cliff recurs across methods [5, 6], with code and math degrading far more than commonsense, so commonsense averages overstate retained coding ability. This positions our work: rather than chase a sub-4-bit win the field has repeatedly failed to find, we test whether *any* in-budget route can beat 4-bit — and report, in §5, that within our search none does.

MoE-aware mixed precision and expert allocation. A line of work allocates bits per expert: rank experts by router-norm change and give rare-but-important experts *more* bits [7]; solve an integer program over $\{2, 3, 4\}$-bit per expert weighted by activation frequency and reconstruction error [5, 6]; or, highest-leverage of all, calibrate the *router* because low-bit quantization perturbs expert *selection* more than expert weights [8]. A cautionary counter-current is that frequency-driven static allocation overfits and can fall below plain GPTQ, and naive expert dropping/merging hurts generative and coding tasks [8, 4]. Our North-aware custom predicate (protect router, global-attention layers and the dense prefix; push experts low) operationalizes this family; we report its outcome in §5. We note that router calibration — the lever this line of work finds highest-leverage [8]— is one we did not run.

Calibrated PTQ: AWQ, GPTQ, and the memory wall. AWQ and GPTQ are the standard strong PTQ backbones [9, 6, 10], with KD-based recovery stacking on top below 4-bit [11, 12, 13]. The practical obstacle on-device is memory: surveys note that mixed-bit and unstructured-sparse speedups are largely CUDA-specific and that on Apple Silicon INT4 is often dequantized for GEMM, making mixed-bit a memory win rather than a throughput win [6, 14]. Crucially, MoE Hessian-based PTQ is memory-bound at scale because per-expert second-order statistics multiply; our streaming GPTQ (§3) attacks exactly this wall.

MoE→dense distillation is expensive and rarely wins. Kim et al. [15] convert a MoE to dense via diversity-aware DO-ACP expert selection plus forward-KL distillation, but find the result still trails a well-trained 1.7B dense model at ∼4B tokens; structured prune-and-distill recovery needs 120–400B tokens [16, 17]. Forward-KL with a shared tokenizer, initialized from the teacher, is the recommended recipe [15, 17]; activating only top-k underuses an MoE teacher [18]. This evidence is precisely why we ran distillation only as a *gated* pilot at a deliberately small budget; our 0/32 result at ∼0.4M tokens is a triage outcome at that budget, not a claim about the ceiling of distillation (§5, §6). Cross-tokenizer KD [19, 20] was out of scope once distillation was a no-go.

Router behavior, expert importance, and on-device runtimes. Router-consistency and expert-collaboration analyses inform which experts are "cold" versus load-bearing for code [21, 22, 14], and router calibration / KD repair can fix router–expert mismatch after surgery [23]. For deployment, mlx-lm / vllm-mlx are the measured fast runtimes on Apple Silicon [24], on-device

MoE scaling is an active target [25], and the Apple Neural Engine underperforms for decode [26]; quantization can also distort the distribution of programming tokens [27]. For agentic evaluation we follow contamination-controlled and harness-fixed practice [28, 29].

3 Method

We describe the compression search space, the streaming GPTQ that makes calibrated PTQ feasible for North, and the gated DO-ACP distillation pilot.

3.1 Compression search space

All quantization keeps North's precision allocation fixed to that of the established 4-bit build: only the expert/MLP linears are quantized (affine, group size 64); the router (`mlp.gate`), attention, embeddings, and norms remain bf16. This both matches the baseline and bounds the second-order memory footprint. Within that envelope we sweep, in increasing effort: (i) uniform bit-width (3/4/5/6-bit) and number format (affine, MXFP4); (ii) built-in mixed-bit predicates (`mixed_2_6`...`mixed_4_8`, a fixed position heuristic, *not* North-aware); (iii) a **custom North-aware per-expert predicate** that protects the router, the 13 global-attention layers, and the dense prefix at high bits while pushing experts to 3-bit, reachable through `mlx_vlm.convert(..., quant_predicate=callable)` without any library fork; and (iv) calibrated streaming GPTQ. Every candidate must pass two hard gates — $\leq$24 GB peak resident at a 32K-token turn and $\geq\sim$20 tok/s decode — and is scored on the HumanEval [30]/MBPP [31] proxy.

3.2 Streaming GPTQ for large MoEs

The strongest calibrated routes (mlx-lm GPTQ/DWQ) became reachable for North only after the `cohere2_moe` architecture could be loaded by mlx-lm; we used a port for that (independently realized upstream [2]) and verified exact logit parity against the reference mlx-vlm implementation (max-abs logit difference 0.0, top-1 / top-5 agreement). But *stock* mlx-lm GPTQ is infeasible on this model on a 48 GB Mac: it accumulates Hessians for all quantizable layers from a single full bf16 forward, which requires the $\sim$60 GB bf16 model resident, and the 128-expert `SwitchLinear` Hessians grow to hundreds of GB.

Our streaming GPTQ is classical sequential GPTQ that never holds more than one layer of bf16 weights at a time. Let $\mathbf{X}_\ell$ be the calibration activations entering layer ℓ. We:

1. load the model lazily ($\sim$0.8 GB RSS; nothing materialized), embed the calibration tokens once into a running hidden state $\mathbf{h}$, and drop the embedding;
2. for each decoder layer in order, materialize *only* that layer's bf16 weights and push $\mathbf{h}$ through North's parallel block $\mathbf{h}' = \mathrm{attn}(\mathrm{norm}(\mathbf{h})) + \mathrm{mlp}(\mathrm{norm}(\mathbf{h})) + \mathbf{h}$, keeping attention and the router in bf16;
3. accumulate per-linear Hessians $\mathbf{H} = \sum_i \mathbf{x}_i \mathbf{x}_i^\top$; for the 128 experts, $\mathbf{H}$ is accumulated *per expert and only over the tokens actually routed to that expert* under top-8 routing — the key step that keeps expert Hessians small rather than 128$\times$ full;
4. GPTQ-quantize [32] this layer's MLP linears with the Cholesky inverse-Hessian column-wise error correction, swap in the quantized modules, recompute the layer output *with the quantized weights* to form $\mathbf{h}$ for the next layer;

5. free the bf16 weights and Hessians and clear the cache before advancing.

Because per-layer bf16 weights, per-expert Hessians, and activations are all freed before the next layer, peak RSS stays flat across depth (§5, Fig. 2). Calibration is code (MBPP splits, 64 samples $\times$ 256 tokens), disjoint by construction from the HumanEval evaluation set. This is a fixed, modest calibration budget: under top-8 routing, 64×256 tokens spread across 128 experts yield *sparse* per-expert Hessians. We did not ablate the calibration size, so a result that calibrated GPTQ *at this budget* ties RTN (§5) shows that our calibration did not help, not that no calibration could.

3.3 Gated DO-ACP distillation pilot

As the only route that could plausibly *beat* a 4-bit model that no compression route had improved on, we ran a cheap, opportunistic MoE→dense gate following Kim et al. [15]. Per MoE layer we score the 128 experts by conditional routing probability (mass-weighted sigmoid gate) $\times$ a weight-based expert output norm, select the top-K, and concatenate their gate/up/down projections into one dense gated FFN (intermediate $K \times 768$), copying attention, embeddings, norms, and the dense prefix unchanged. We cache the teacher's top-k logits *before* the in-place conversion (so a second model copy is never resident), then train the rebuilt FFNs with forward-KL against those fixed targets. The gate's question is narrow: at a feasible budget, does this *credibly trend* toward beating ∼0.89 HumanEval? The pilot used $K = 8$ (∼3.29B dense student) and ∼0.38M KD tokens — roughly 260–790× below the 0.1–0.3B target and orders of magnitude below the literature budget where MoE→dense still trailed small pretrained models. It is a trajectory probe, not a recovered model: the gate tests whether a cheap run trends upward, not whether distillation could ever succeed.

4 Experimental Setup

Data. This is a model-efficiency study with no supervised training table. Three data buckets matter, all derived from non-evaluation data and held strictly disjoint from the evaluation suites (problem-id and repo-level deduplication; SWE-Bench Verified [33] source repos excluded from any code corpus): (a) *evaluation* — HumanEval and MBPP (the proxy) and a seeded SWE-Bench Verified subset (the only agentic data point); (b) *calibration* for PTQ — code (MBPP splits), 64 samples $\times$ 256 tokens for streaming GPTQ; (c) *KD data* for the distillation pilot — North's own top-k logits over a small MBPP code corpus. No Qwen3.6 data was used (the distillation arm was a no-go), so there is no attribution constraint on any released artifact.

Proxy protocol. HumanEval/MBPP base pass@1, generated with mlx-vlm and scored by running each problem's canonical `check()` in a subprocess. North is a reasoning model, so the harness isolates the final answer after the `<|START_TEXT|>` marker and concatenates code blocks. Decoding is greedy with `max_tokens`=4096; a downward bias arises from 15–22 truncations per run when the reasoning trace overruns the budget (this bias is not equal across configs — e.g. RTN truncates 15 on HumanEval vs. streaming-GPTQ's 18, §5). Memory is measured as peak RSS with *real* code prompts (dummy prompts under-report MoE memory), including a real 32,034-token prompt + 1024-token generation to represent a 32K agentic turn.

Agentic protocol. We use a seeded, stratified SWE-Bench Verified subset ($n = 30$, seed 1234, 12 repositories). North's native multi-turn agent expects a Dockerized `/workspace` unavailable on the Mac without a tunnel, so we fall back to the sanctioned *single-shot* variant: the prompt provides the issue plus oracle file contents, and the model returns SEARCH/REPLACE edit blocks. We apply the blocks and let `git diff` produce the patch. Decoding is greedy (+ repetition penalty 1.1), `max_tokens`=9000. Generation runs on the Mac (4-bit MLX); test execution uses the *official* swebench Docker harness on a Lambda Linux/x86 GPU instance. This single-shot, oracle-file scaffold is an explicit lower bound relative to a real agent loop.

Gates. Every finalist must satisfy the hard ≤24 GB resident cap at a 32K turn and the ≥∼20 tok/s decode floor. The promotion guard for a *win* on the proxy was set in Phase 0 to the measured noise floor: pooled binomial SE ≈1.46%, so $W \approx 1.5\%$ plus a McNemar test.

5 Results

All reported numbers trace to recorded run artifacts. We organize the results around the central question: can anything beat RTN 4-bit at ≤24 GB?

Baseline and headline. RTN 4-bit North scores 0.8902 (146/164, 15 truncated) on full HumanEval base and 0.9066 (233/257, 22 truncated) on MBPP base (sanitized), for a pooled pass@1 of 0.9002 (379/421). It uses 19.33 GB peak at short context and, at a real 32K-token turn, 20.81 GB peak with 40.2 tok/s decode and 410 tok/s prefill — inside the 24 GB cap with ∼3 GB of headroom and well above the ∼20 tok/s floor. Memory was never the binding constraint.

The quantization sweep: nothing beats 4-bit. Table 1 summarizes the search. Most rows were scored on the HumanEval-50 screening subset (per-arm binomial SE ≈±5 pts), adequate for triage but not for ranking near the ceiling; we do *not* read subset accuracy as a win/loss against the full-suite baseline. Crucially, the success bar is accuracy at ≤24 GB, *not* minimal memory — so a higher-bit config that fits the budget cannot be dismissed for merely using more memory than 4-bit. The single most threatening candidate is therefore `mixed_4_8` (5.21 average bits/weight; 20.5 GB peak, inside the budget) — the highest-bit configuration that fits, and the one most likely to recover any accuracy 4-bit gives up. We promoted it to a full HumanEval-164 run (local, ∼0 GPU-hr). It scores 0.9024 (148/164, 15 truncated): nominally +2 problems over RTN's 0.8902, but a paired McNemar gives exact $p = 0.7905$ (both pass 140; `mixed_4_8`-only 8, RTN-only 6), and the +1.2% gap is below both the W=1.5% promotion guard and the 2.3% HumanEval-164 binomial SE. The two are statistically indistinguishable; moreover, all 6 problems RTN passes and `mixed_4_8` misses are `mixed_4_8` *truncations* — the generation-length confound (§6), not weaker code. Spending an extra ∼1.2 GB and ∼0.7 bits/weight on the experts buys no measurable coding accuracy. The uniform 5-bit config is excluded on a harder ground: at 22.82 GB short-context peak it exceeds the 24 GB budget once the 32K KV cache is added (≈24.3 GB, using RTN's measured +1.48 GB short→32K delta), so it fails the memory gate at the context lengths the deliverable targets. At the low end, `mixed_2_6` collapses entirely (0.020, 50/50 truncated — a hard floor at 2-bit experts). The one memory-*saving* config we re-ran at full scale, `mixed_3_4` (∼3.5 avg bits), scores 0.811 on full HumanEval-164 at `max_tokens`=4096 (133/164, 28 truncated) and 0.8598 at 8192 (141/164, 20 truncated) — genuinely below RTN, a real accuracy-for-memory tradeoff. The

Table 1: Quantization sweep. Coding eval is full HumanEval-164 base pass@1 where available, else the HumanEval-50 *screening* subset (per-arm SE ≈±5 pts; subset and full-suite numbers are *not* directly comparable). Peak GB is short-context generation peak unless the 32K-turn value is given. The three full-suite configs (RTN, `mixed_4_8`, streaming-GPTQ) are mutually statistically indistinguishable (McNemar $p = 0.79$ and $p = 1.000$): no configuration beats RTN 4-bit on coding eval by the W=1.5% margin. The in-budget higher-bit `mixed_4_8` ties on the full suite; uniform 5-bit fails the 24 GB gate once the 32K KV cache is included. †Streaming-GPTQ shares RTN's exact 4-bit allocation (expert/MLP-only, affine g64; 18 GB / 736-tensor checkpoint), so its generation footprint and throughput equal the RTN row by construction; only its quantization-time memory (3.75 GB, Fig. 2) was separately profiled.

Config	Coding eval	Peak GB	tok/s	Note
RTN 4-bit (baseline)	0.8902 (HE-164)	19.33 (20.81@32K)	58.1 (40.2@32K)	baseline
`mixed_4_8`	0.9024 (HE-164)	20.53 (∼22@32K)	75.6	**tie** (p=0
streaming-GPTQ 4-bit	0.8841 (HE-164)	19.33†	58.1†	**tie** (p=1
5-bit	0.900 (HE-50)	22.82	54.0	over 24 G
`mixed_3_4`	0.811/0.8598 (HE-164 @4096/8192)	14.81	84	smaller,
`mixed_3_6`	0.860 (HE-50, screen)	16.16	—	screenin
custom per-expert	0.880 (HE-50, screen)	12.68	80.5	3.568 bit
`mixed_2_6`	0.020 (HE-50, screen)	13.02	—	collapse

North-aware custom predicate reaches 3.568 effective bits/weight at 12.68 GB; on the screening subset it is indistinguishable from `mixed_3_4` and was not promoted. Figure 1 plots the frontier.

Streaming GPTQ: feasible, and a within-noise tie. The streaming quantizer ran the full 49-layer model at 3.75 GB peak in ∼70 minutes, producing an 18 GB / 736-tensor 4-bit checkpoint; a 3-layer profile peaks at 2.13 GB, and RSS stays flat across depth (Fig. 2) — the exact failure mode that blocks stock GPTQ (whole model plus all-layer Hessians, hundreds of GB) is eliminated. On accuracy, calibrated streaming-GPTQ scores 0.8841 (145/164, 18 truncated) on full HumanEval vs. RTN's 0.8902 (146/164, 15 truncated). A paired McNemar analysis over the 164 problems gives a contingency of both-correct 137, RTN-only 9, streaming-only 8, neither 10 (17 discordant pairs), for an exact 2-sided $p = 1.000$: the two are statistically indistinguishable. The paired difference is also inside our pre-set equivalence margin (the HE-only binomial SE is ≈2.4% at $p \approx 0.89$, $n = 164$, against the pooled $W \approx 1.5\%$ on $n = 421$), so this is equivalence within $\pm W$, not merely a failure to reject. The small marginal deficit is moreover a truncation artifact, not worse code: of the 9 problems RTN passes but streaming-GPTQ fails, 7 were streaming-GPTQ truncations (the model looped past the token budget), consistent with streaming-GPTQ's higher truncation count (18 vs. RTN's 15). The contribution here is the feasibility (a genuine MoE-quantization instrument) and the negative finding it certifies — at our calibration budget, code-calibrated GPTQ adds nothing measurable over plain RTN for this model.

Distillation gate: no-go. The DO-ACP MoE→dense pilot produced a 3.29B dense student and trained its FFNs with forward-KL (187 steps, ∼0.38M tokens; KD loss 2.62 → 1.47), but scored 0/32 on HumanEval base (one-sided 95% upper bound ≈9% by the rule of three) — a flat floor, not a trend toward 0.89. Total paid compute for the gate was ∼0.53 GPU-hours. This is a triage

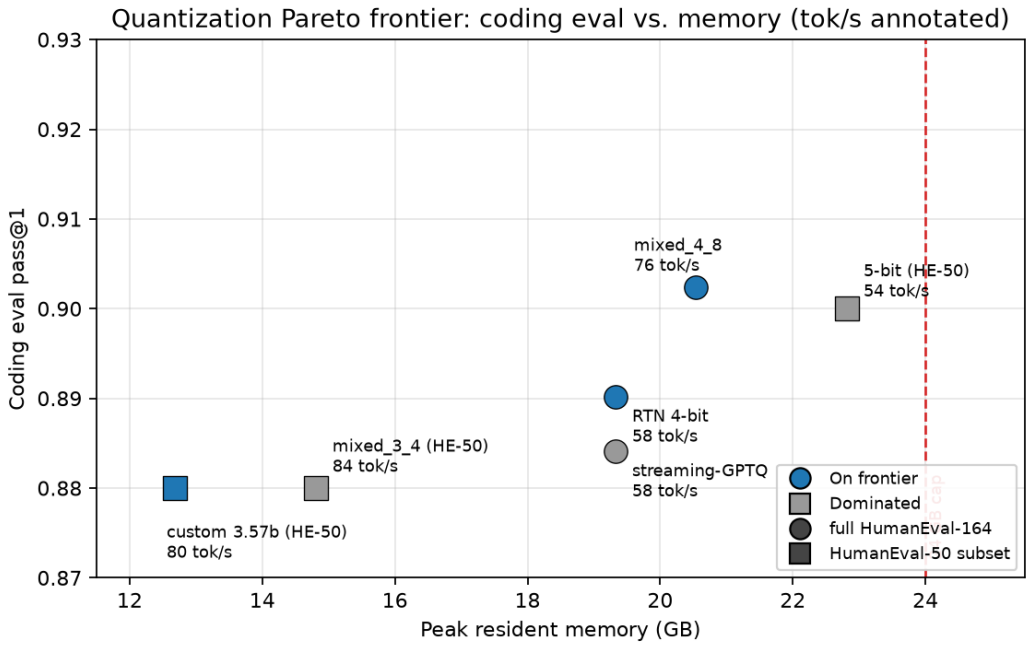

Figure 1: Coding-eval vs. peak memory, tok/s annotated. RTN 4-bit, `mixed_4_8`, and calibrated streaming-GPTQ (full HumanEval-164, circles) coincide within noise ($p \geq 0.79$); the remaining squares are HumanEval-50 subset points and are not directly comparable to the full-suite numbers. The takeaway: no route exceeds 4-bit accuracy by the W=1.5% margin, and all in-budget configs sit inside the 24 GB cap (uniform 5-bit, not shown as viable, exceeds it at 32K context).

outcome: at this deliberately small budget, FFN-only KD shows no upward trend, which is the only question the gate was built to answer. It is *not* evidence about the ceiling of MoE→dense distillation at an adequate budget; the literature already finds that regime needs far more tokens than we spent. The arm was closed as not-cheaply-promising, not as impossible.

Agentic characterization: under our scaffold, the limiter is generation stability. On the seeded SWE-Bench Verified subset ($n = 30$), 4-bit North resolved 6/30 = 20.0% (Wilson 95% CI [9.5%, 37.3%]). Under our single-shot scaffold the observed limiter is not edit quality but *non-termination*: 21/30 instances looped to the 9000-token cap, of which 20 produced an empty patch (one looped but still yielded a non-empty patch); with 6 resolved, that leaves 24 misses, 20 of them empty patches. A salient looping trigger was that the model loops when asked to hand-count unified-diff line numbers, which is why our scaffold takes SEARCH/REPLACE blocks rather than raw diffs. *Conditional on emitting a patch*, the model resolves 6/10 = 60% (Wilson CI [31%, 83%]), including an exact gold match (requests-2317); we report this only as an **illustrative, selection-biased small-n number** — conditioning on the patches that were emitted is not a clean capability estimate. We caution that, with no agentic bit-width contrast in our experiments, we cannot attribute this looping to quantization; it is the limiter we *observe* under this scaffold, full stop. The relation to the published 67.6 reference figure is deferred to §6 as an explicitly non-comparable caveat.

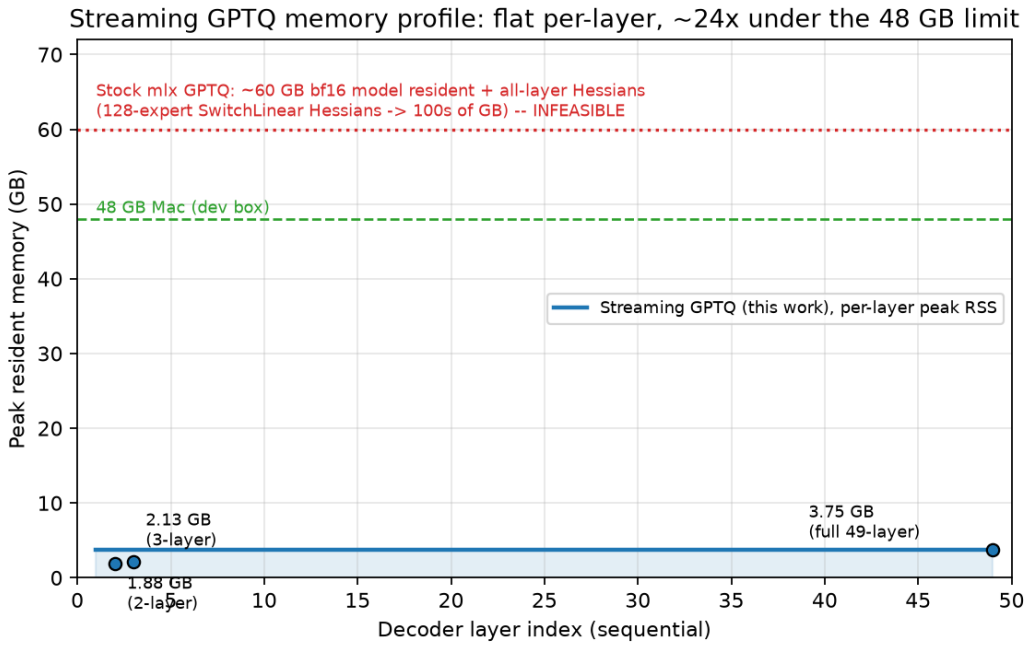

Figure 2: Streaming-GPTQ peak RSS is flat across the 49 layers (full run 3.75 GB; 3-layer profile 2.13 GB), ∼24× under the 48 GB Mac. Stock mlx-lm GPTQ requires the ∼60 GB bf16 model resident plus all-layer Hessians (the 128-expert `SwitchLinear` Hessians reaching hundreds of GB) — infeasible on this hardware.

Outcome: answering the title. Is 4-bit the ceiling? Within everything we tested, *yes*: no compression route that fits the ≤24 GB budget beats RTN 4-bit by our margin — not the cheaper ones, and not the in-budget higher-bit `mixed_4_8`, which we promoted to a full HumanEval-164 run precisely because it was the likeliest to refute the claim and which ties ($p = 0.79$, $+1.2\% < W$). The calibrated streaming route is likewise a tie ($p = 1.000$, within $\pm W$), and the distillation gate failed at its budget. Against our stated goal of *beating* 4-bit, that affirmative answer is precisely the negative result — the two framings are the same fact. The secondary deliverable (a documented quant-vs-distill Pareto at 24 GB) is met, and the best openly-runnable ≤24 GB North *is* the 4-bit model.

6 Discussion and Limitations

The honest reading is that for North-Mini-Code, within the routes we searched and inside the vendor's fixed bf16-router/attention envelope, no route that fits the ≤24 GB budget beats 4-bit — including the in-budget higher-bit `mixed_4_8`, which we tested at full scale and which ties ($p = 0.79$). 4-bit is the ≤24 GB ceiling we could find. We scope this carefully. Our highest-precision *measured* configuration is `mixed_4_8` at 5.2 average bits/weight; we measured no full 8-bit or bf16 reference (bf16 generation OOMs the 48 GB Mac, an 8-bit reference is future work), so we cannot establish loss versus full precision and do *not* claim 4-bit is "lossless" — only that nothing we tried beat it. The "ceiling" claim is likewise bounded: we held the router, attention, embeddings, and norms at bf16 to match the baseline, and we did not run router calibration, which the MoE literature

Table 2: Headline outcomes against the project gates. The primary criterion (beat 4-bit by $\geq W$) was **not met** — the two most direct threats, the in-budget higher-bit `mixed_4_8` and calibrated streaming-GPTQ, are both statistical ties on full HumanEval-164 ($p = 0.79$ and $p = 1.000$); the memory and throughput gates are both met. The deliverable is therefore the 4-bit model itself (`mixed_4_8` matches it within noise but costs ∼1.2 GB more for no measurable accuracy gain). The published 67.6 reference is *not* listed here as a comparison: it uses different precision and a real agent harness, and is discussed only as a non-comparable caveat in §6.

Metric	Result (4-bit North)	Gate
Proxy pass@1 (pooled HE+MBPP)	0.9002 (379/421)	beat 4-bit by W≈1.5%: **not met (tie)**
Peak resident @ 32K turn	20.81 GB	$\leq$ 24 GB: **met**
Decode throughput @ 32K	40.2 tok/s	$\geq\sim$20 tok/s: **met**
SWE-Bench Verified subset	20.0% (6/30), CI [9.5, 37.3]	characterization only (no gate)
conditional on a patch	60% (6/10), CI [31, 83]	illustrative, selection-biased

identifies as the highest-leverage lever [8, 23]; a different envelope could move the result. With those scopes, the finding still matches the strongest prior on the near-twin architecture [4]: at 4-bit you are already at the knee, and below it code collapses. Memory was never binding (∼3 GB of headroom at 32K), and throughput cleared its floor by 2×.

The practically important finding is elsewhere: under our single-shot scaffold, the limiter on *agentic* coding is generation-length stability. Across the proxy and agentic suites the dominant failure is non-termination / repetition looping (HumanEval 15/164, MBPP 22/257, agentic 21/30 truncations), not wrong logic — when the model terminates it is largely correct (∼89–91% proxy; 60% agentic, the latter an illustrative selection-biased small-n figure). A small repetition penalty (1.1) noticeably reduced the repetition-looping seen under pure greedy decoding but did not eliminate it. This points to decoding and stop-criteria tuning, not bit allocation, as the highest-leverage next step. We deliberately make no causal claim that this instability is a quantization artifact: we ran no agentic bit-width contrast, so we cannot separate any bit-width contribution from the scaffold and decoding settings.

Two artifacts have independent value despite the negative headline: the streaming GPTQ makes Hessian-based PTQ feasible for 100+-expert MoEs on commodity unified memory, and the study itself is a clean, reproducible characterization of why no in-budget route we tried beats 4-bit for this model class.

6.1 Threats to Validity

Construct. The proxy is HumanEval/MBPP *base*, not the "+" suites, and measures single-shot function completion, not the agentic objective; it may overstate retained agentic ability, which is why we add the SWE-Bench subset. **Internal.** The agentic number uses a single-shot, oracle-file SEARCH/REPLACE scaffold, which lower-bounds a real multi-turn agent; greedy decoding with a fixed token cap turns the model's looping tendency directly into failed instances, so the 20% conflates capability with a decoding pathology (hence we report the conditional 60%, itself an illustrative selection-biased small-n figure). Truncations impose a downward bias on every proxy number too, and that bias is *not* equal across configs (RTN 15 vs. streaming-GPTQ 18 on HumanEval), which alone accounts for 7 of the 9 problems on which RTN beats streaming-GPTQ

— a confound, not a code-quality gap. **External / statistical.** The agentic subset is $n = 30$ with a wide Wilson CI ($[9.5\%, 37.3\%]$), adequate for characterization but not for a precise claim; results are for one model on one runtime (MLX) under one memory budget. **No full-precision reference.** Our highest-precision measured configuration is `mixed_4_8` (5.2 average bits/weight); we did not measure a full 8-bit or bf16 reference (bf16 generation OOMs the 48 GB development Mac, 8-bit is future work). We therefore cannot quantify loss versus full precision and make no "lossless" claim — our finding is strictly that nothing we tried beat 4-bit in our search. **Search/envelope scope.** We held router, attention, embeddings, and norms at bf16 to match the baseline and did not run router calibration (the literature's highest-leverage MoE lever [8, 23]); the GPTQ result is at a single fixed calibration budget (64×256 tokens, sparse per-expert Hessians) that we did not ablate, so it shows our calibration did not help, not that none can. **Reference comparison.** The published 67.6 on SWE-Bench Verified uses different precision and a real multi-turn agent harness; our 20.0% is a lower bound under a much weaker single-shot oracle-file scaffold on a small subset, so the two are explicitly not apples-to-apples and we do not read a $67.6 \rightarrow 20$ capability collapse into them. **Distillation scope.** The no-go is a triage outcome at a deliberately cheap gate (FFN-only KD, frozen attention/embeddings, $K = 8$, $\sim$0.4M tokens, $n = 32$ eval; one-sided 95% upper bound $\approx$9%); the gate's question was only whether it cheaply *trends* toward beating 0.89, and a 0/32 floor answers no — it is not evidence about distillation's ceiling at an adequate budget.

7 Availability

A GitHub companion repository (`https://github.com/transformerlab/exp-north-mini-code-mac`) provides the streaming-GPTQ code, the evaluation harness (HumanEval/MBPP plus the SWE-Bench Verified subset scripts and seeds), and the convert / reproduction recipes. The `cohere2_moe` architecture support for mlx-lm is available via the concurrent upstream pull request [2]; we used an independent port only as a study enabler. The 4-bit weights are the existing public build `mlx-community/North-Mini-Code-1.0-4bit`. No datasets are released. No Qwen3.6 data was used — the distillation arm was a no-go — so no third-party attribution constraint applies to any released artifact. The model and its quantizations are Apache-2.0.

8 Conclusion and Future Work

Is 4-bit the ceiling? For North-Mini-Code-1.0 on Apple Silicon, within what we tested the answer is yes, and against our goal of beating it that is a negative result: no in-budget quantization route and no cheap distillation beats RTN 4-bit on coding eval — the two closest, the in-budget higher-bit `mixed_4_8` and calibrated streaming-GPTQ, are both statistical ties on full HumanEval-164 ($p = 0.79$, $p = 1.000$, within $\pm W$) — and under our single-shot scaffold the practical agentic limiter is generation-length instability rather than bit-width. We measured no full-precision reference, so this is a ceiling within our search and the vendor's fixed bf16-router/attention envelope, not a loss-vs-full-precision claim. The 4-bit model fits in 20.81 GB at a 32K turn and decodes at 40.2 tok/s. Independently, we contribute a streaming GPTQ that makes Hessian-based PTQ feasible for large MoEs on commodity unified memory. Ranked next steps: (1) attack the real limiter — decoding and stop-criteria tuning (repetition penalties, dynamic stopping, length-aware budgets) to suppress repetition loops; (2) run North's *native* multi-turn agent loop in Docker over the network for a precise agentic number; (3) tighten the agentic estimate with the full 500-instance SWE-Bench

Verified run; and (4) revisit distillation at a budget large enough to be informative — our gate only established that the cheap regime does not trend upward, not that the approach is ruled out.

References

[1] Cohere Labs. North-Mini-Code-1.0. Hugging Face, CohereLabs/North-Mini-Code-1.0, 2025. URL https://huggingface.co/CohereLabs/North-Mini-Code-1.0.

[2] mlx-lm contributors. [Cohere] Add cohere2_moe model support. Pull request #1340, ml-explore/mlx-lm, GitHub, 2026. URL https://github.com/ml-explore/mlx-lm/pull/1340. Concurrent upstream support for the cohere2_moe architecture.

[3] Qwen Team. Qwen3 Technical Report, 2025. URL https://arxiv.org/abs/2505.09388.

[4] Mike Lasby, Ivan Lazarevich, Nish Sinnadurai, Sean Lie, Yani Ioannou, and Vithursan Thangarasa. REAP the Experts: Why Pruning Prevails for One-Shot MoE Compression, 2025. URL https://arxiv.org/abs/2510.13999.

[5] Wei Huang, Yue Liao, Jianhui Liu, Ruifei He, Haoru Tan, Shiming Zhang, Hongsheng Li, Si Liu, and Xiaojuan Qi. Mixture Compressor for Mixture-of-Experts LLMs Gains More, 2024. URL https://arxiv.org/abs/2410.06270.

[6] Mariam Rakka, Marios Fournarakis, Olga Krestinskaya, Jinane Bazzi, Khaled N. Salama, Fadi Kurdahi, Ahmed M. Eltawil, and Mohammed E. Fouda. Mixed-Precision Quantization for Language Models: Techniques and Prospects, 2025. URL https://arxiv.org/abs/2510.16805.

[7] Mohammed Nowaz Rabbani Chowdhury, Kaoutar El Maghraoui, Hsinyu Tsai, Naigang Wang, Geoffrey W. Burr, Liu Liu, and Meng Wang. Efficient Quantization of Mixture-of-Experts with Theoretical Generalization Guarantees, 2026. URL https://arxiv.org/abs/2604.06515.

[8] Yuanteng Chen, Yuantian Shao, Peisong Wang, and Jian Cheng. EAC-MoE: Expert-Selection Aware Compressor for Mixture-of-Experts Large Language Models, 2025. URL https://arxiv.org/abs/2508.01625.

[9] Yutong Liu, Cairong Zhao, and Guosheng Hu. A Comprehensive Evaluation on Quantization Techniques for Large Language Models, 2025. URL https://arxiv.org/abs/2507.17417.

[10] Guanxi Lu, Hao Chen, Zhiqiang Que, Wayne Luk, and Hongxiang Fan. Enhancing Trustworthiness with Mixed Precision: Benchmarks, Opportunities, and Challenges, 2025. URL https://arxiv.org/abs/2511.22483.

[11] Xun Wu, Shaohan Huang, Wenhui Wang, Ting Song, Li Dong, Yan Xia, and Furu Wei. BitNet Distillation, 2025. URL https://arxiv.org/abs/2510.13998.

[12] Jung Hyun Lee, Seungjae Shin, Vinnam Kim, Jaeseong You, and An Chen. Unifying Block-wise PTQ and Distillation-based QAT for Progressive Quantization toward 2-bit Instruction-Tuned LLMs, 2025. URL https://arxiv.org/abs/2506.09104.

[13] Fangxin Liu, Ning Yang, Junping Zhao, Tao Yang, Haibing Guan, and Li Jiang. LCD: Advancing Extreme Low-Bit Clustering for Large Language Models via Knowledge Distillation, 2025. URL https://arxiv.org/abs/2506.12038.

[14] Yanyue Xie, Zhi Zhang, Ding Zhou, Cong Xie, Ziang Song, Xin Liu, Yanzhi Wang, Xue Lin, and An Xu. MoE-Pruner: Pruning Mixture-of-Experts Large Language Model using the Hints from Its Router, 2024. URL https://arxiv.org/abs/2410.12013.

[15] Junhyuck Kim, Jihun Yun, Haechan Kim, Gyeongman Kim, Joonghyun Bae, and Jaewoong Cho. Pruning and Distilling Mixture-of-Experts into Dense Language Models, 2026. URL https://arxiv.org/abs/2605.28207.

[16] Zichong Li, Chen Liang, Zixuan Zhang, Ilgee Hong, Young Jin Kim, Weizhu Chen, and Tuo Zhao. SlimMoE: Structured Compression of Large MoE Models via Expert Slimming and Distillation, 2025. URL https://arxiv.org/abs/2506.18349.

[17] Shengkun Tang, Zekun Wang, Bo Zheng, Liangyu Wang, Rui Men, Siqi Zhang, Xiulong Yuan, Zihan Qiu, Zhiqiang Shen, and Dayiheng Liu. SlimQwen: Exploring the Pruning and Distillation in Large MoE Model Pre-training, 2026. URL https://arxiv.org/abs/2605.08738.

[18] Gyeongman Kim, Gyouk Chu, and Eunho Yang. Every Expert Matters: Towards Effective Knowledge Distillation for Mixture-of-Experts Language Models, 2025. URL https://arxiv.org/abs/2502.12947.

[19] Sharath Turuvekere Sreenivas, Adithyakrishna Venkatesh Hanasoge, Mingyu Yang, Ali Taghibakhshi, Saurav Muralidharan, Ashwath Aithal, and Pavlo Molchanov. X-Token: Projection-Guided Cross-Tokenizer Knowledge Distillation, 2026. URL https://arxiv.org/abs/2605.21699.

[20] Buu Phan, Ashish Khisti, and Karen Ullrich. Cross-Tokenizer Likelihood Scoring Algorithms for Language Model Distillation, 2025. URL https://arxiv.org/abs/2512.14954.

[21] Jingcong Liang, Siyuan Wang, Miren Tian, Yitong Li, Duyu Tang, and Zhongyu Wei. Not All Models Suit Expert Offloading: On Local Routing Consistency of Mixture-of-Expert Models, 2025. URL https://arxiv.org/abs/2505.16056.

[22] Yuanbo Tang, Yan Tang, Naifan Zhang, Meixuan Chen, and Yang Li. Unveiling Hidden Collaboration within Mixture-of-Experts in Large Language Models, 2025. URL https://arxiv.org/abs/2504.12359.

[23] Sieun Hyeon and Jaeyoung Do. Is Retraining-Free Enough? The Necessity of Router Calibration for Efficient MoE Compression, 2026. URL https://arxiv.org/abs/2603.02217.

[24] Wayner Barrios. Native LLM and MLLM Inference at Scale on Apple Silicon, 2026. URL https://arxiv.org/abs/2601.19139.

[25] Yanbei Chen, Hanxian Huang, Ernie Chang, Jacob Szwejbka, Digant Desai, Zechun Liu, Vikas Chandra, and Raghuraman Krishnamoorthi. MobileMoE: Scaling On-Device Mixture of Experts, 2026. URL https://arxiv.org/abs/2605.27358.

[26] Ramchand Kumaresan. Orion: Characterizing and Programming Apple's Neural Engine for LLM Training and Inference, 2026. URL https://arxiv.org/abs/2603.06728.

[27] Viacheslav Siniaev, Iaroslav Chelombitko, and Aleksey Komissarov. Compressed code: the hidden effects of quantization and distillation on programming tokens, 2026. URL https://arxiv.org/abs/2601.02563.

[28] Smriti Jha, Matteo Paltenghi, Chandra Maddila, Vijayaraghavan Murali, Shubham Ugare, and Satish Chandra. REAP: Automatic Curation of Coding Agent Benchmarks from Interactive Production Usage, 2026. URL https://arxiv.org/abs/2604.01527.

[29] Lilin Wang, Lucas Ramalho, Alan Celestino, Phuc Anthony Pham, Yu Liu, Umang Kumar Sinha, Andres Portillo, Onassis Osunwa, and Gabriel Maduekwe. SWE-Bench++: A Framework for the Scalable Generation of Software Engineering Benchmarks from Open-Source Repositories, 2025. URL https://arxiv.org/abs/2512.17419.

[30] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating Large Language Models Trained on Code, 2021. URL https://arxiv.org/abs/2107.03374.

[31] Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program Synthesis with Large Language Models, 2021. URL https://arxiv.org/abs/2108.07732.

[32] Elias Frantar, Saleh Ashkboos, Torsten Hoefler, and Dan Alistarh. GPTQ: Accurate Post-Training Quantization for Generative Pre-trained Transformers, 2022. URL https://arxiv.org/abs/2210.17323.

[33] Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. SWE-bench: Can Language Models Resolve Real-World GitHub Issues?, 2023. URL https://arxiv.org/abs/2310.06770.