What Happens When AI Starts Improving AI?

Astra shows how far AI has come. What takes it further?

The continued progress of today's most capable models depends on **smarter decisions before training, better strategies during training, and more efficient infrastructure underneath both**—work still largely led by human researchers. When AI can pursue these improvements recursively, the pace of progress changes fundamentally: each verified advance strengthens the next cycle.

This is why we built Apex Intelligence's automated AI research system for discovering and validating improvements across the AI training stacks.

In its first evaluation, our system achieved three new official SOTAs on SimpleTES, GPUMode TriMul (H100), and MLS-Bench Fused Causal Attention, with its TriMul result surpassing TTT-Discover from a team spanning Stanford, NVIDIA, and other institutions. On NanoChat Autoresearch, its B200 result outperformed published results from both Recursive SuperIntelligence and Tencent Hunyuan's Hyra, placing it near the public SOTA for fixed-budget LLM training.

This demonstrates that Apex's automated recursive-research paradigm matches or outperforms top-tier human engineering and state-of-the-art automated systems across **model-scaling prediction, fixed-budget language-model training, and GPU kernel performance.**

Within each benchmark, the research cycle runs autonomously: the system identifies what to improve, tests its ideas, verifies the results, and learns from the evidence. These frontier-level outcomes create stronger starting points for the cycles that follow—this is how improvement becomes recursive.

Benchmark	Task Description	Prior Frontier Results	Apex	Achievements
SimpleTES SLD Bench	Infers symbolic scaling laws from experimental data, scored by extrapolation accuracy on held-out scaling regimes.	0.8613 average score	**0.8846 average score**	2.70% higher average score than the previous SOTA
NanoChat Autoresearch	Improves a small language model by editing its training code within a fixed five-minute, single-GPU budget, measured by val_bpb (lower is better).	Public SOTA AutoTrust AI's ScienceGuru: 0.889522 Recursive Best single run: 0.903891 Tencent Hunyuan's Hyra: 0.901543	**0.892426 val_bpb**	Within 0.002904 val_bpb of public SOTA 0.011465 lower than Recursive 0.009117 lower than Hyra
GPU MODE TriMul (H100)	Optimizes the Triangle Multiplicative Update forward pass on an H100, measured by geometric-mean latency across seven configurations (lower is better).	stashuk-olek 1,064.9 µs	**1,036.1 µs**	2.71% lower latency than the previous SOTA
MLS-Bench: Fused Causal Attention Kernel	Builds a correct OpenAI Triton fused causal-attention kernel for H100, measured by throughput across three configurations (higher is better).	338.7 / 405.1 / 385.7 TFLOPS (hdim 64 / 128 / 256)	**430.7 / 468.4 / 451.7 TFLOPS** (hdim 64 / 128 / 256)	27.18% / 15.63% / 17.11% higher throughput than the previous SOTA (hdim 64 / 128 / 256)

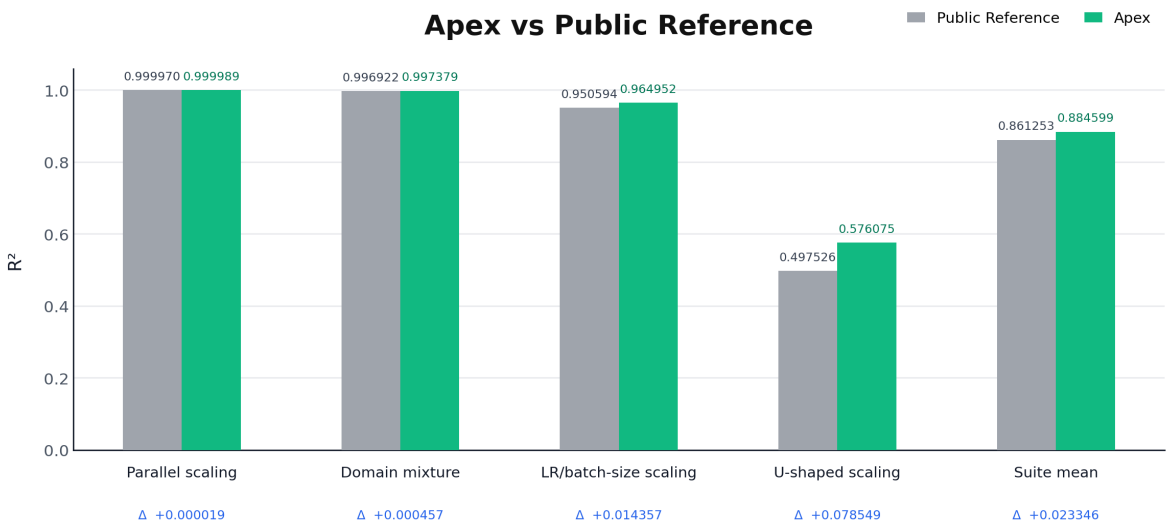
Higher is better for SimpleTES and MLS-Bench; lower is better for NanoChat and TriMul.
NanoChat comparisons use 300-second, single-B200, single-run results.

Evaluation I: SLDBench (SimpleTES 4-Task Protocol)

Before a large training run begins, researchers must predict performance at scales they cannot afford to test directly. Scaling laws guide decisions about model size, data composition, learning rate, batch size, and compute; a law that fits small experiments but extrapolates poorly can misdirect an entire training budget.

SLDBench turns this uncertainty into a measurable discovery task built from more than **5,000 published LLM training experiments**. Each system must produce a symbolic law and a fitting procedure that generalize to held-out scaling regimes. Following the four-task evaluation used by SimpleTES, our AI system is tested on **parallel scaling, domain-mixture scaling, learning-rate/batch-size co-scaling, and U-shaped compute scaling**.

Under the pinned public evaluators, our system improves the previous public reference on every task, raising the four-task average from **0.8613 to 0.8846 (+2.71%)**. The largest gain comes on U-shaped scaling, where the score rises from **0.4975 to 0.5761 (+15.80%)**. Learning-rate/batch-size co-scaling improves from **0.9506 to 0.9650 (+1.51%)**, with further gains on parallel and domain-mixture scaling. These scores measure performance on held-out regimes, rewarding laws that remain accurate beyond the experiments used to fit them.



The clearest method-level advance appears on U-shaped compute scaling. Our AI system represents the curve as the sum of a persistent scaling trend and a localized regime effect that emerges over a limited compute range and then decays. This structure prevents a temporary reversal in the observed data from being projected indefinitely.

The distinction becomes decisive when the evaluator withholds the curve's final phase. Flexible equations can reproduce the observed bend while extrapolating in the wrong direction; explicitly modeling the onset, duration, and decay of the transient component produces more accurate post-turning-point predictions.

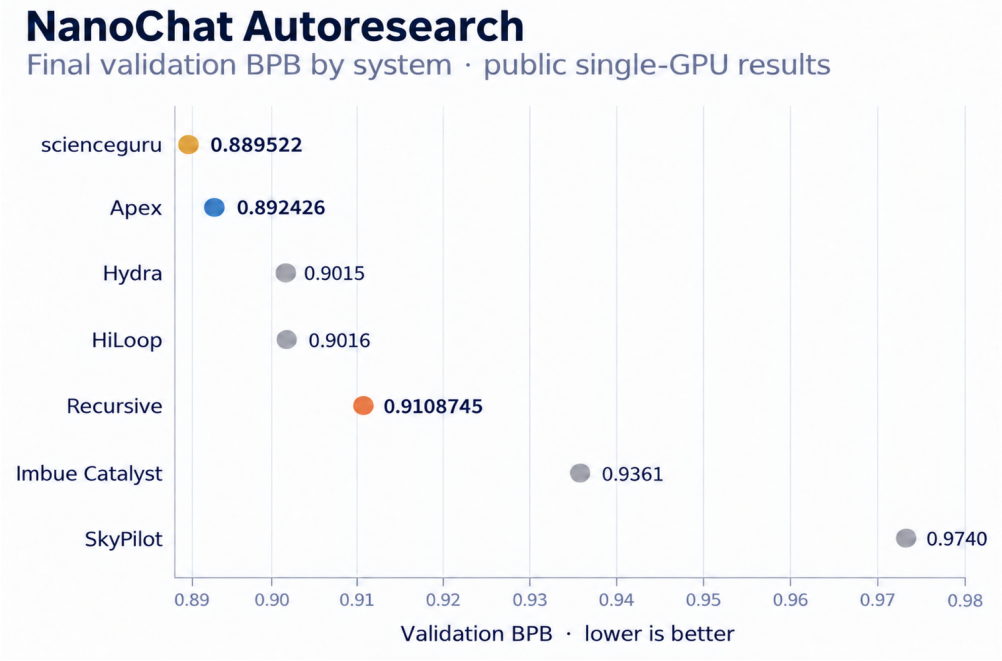
The result is technically meaningful because the decomposition improves held-out accuracy while remaining interpretable and testable. It suggests a reusable hypothesis for scaling-law discovery: an apparent reversal may reflect a temporary regime layered over a longer-term scaling trend.

Evaluation II: NanoChat Autoresearch

Fixed-budget training forces every idea to earn its place: improvements must come from using the same time and hardware more effectively. This makes NanoChat Autoresearch a direct test of whether an AI system can turn rapid experimentation into a stronger model-training implementation.

Released by Andrej Karpathy in March 2026, NanoChat Autoresearch gives an agent one GPU and the freedom to modify the model architecture, optimizer, hyperparameters, and training code. Every experiment receives the same five-minute budget and is scored by validation bits per byte (`val_bpb`; lower is better). The setup has since expanded into autoresearch@home, where human participants and AI agents share experiments and compete against the strongest public solutions.

Under the fixed 300-second single-B200 setting, our AI system reaches **0.892426 val_bpb**, placing it near the public frontier. This surpasses Recursive SuperIntelligence's June 2026 best published single run of **0.903891** and 10-seed mean of **0.9108745**, as well as Tencent Hunyuan's July 2026 Hyra result of **0.901543**. It approaches the public leaderboard leader as of September 2026, AutoTrust AI's ScienceGuru, at **0.889522**.



The decisive improvement comes from carrying n-gram sparsity through the training computation. The ScienceGuru implementation demonstrates the value of large n-gram memories, yet each batch accesses only a small fraction of their rows. Our AI system identifies dense gradient costs that remain in the training path and replaces them with a compact backward pass that updates only the rows used by the current batch.

This change cuts peak training memory from approximately **177.7 GB to 140.6 GB—a 20.9% reduction—while preserving near-SOTA quality**. Within the same 300-second budget, our AI system reaches **0.892426 val_bpb** while processing approximately **4.5% fewer tokens** than the ScienceGuru implementation, showing that near-SOTA quality does not depend on higher token throughput.

Our AI system extends the same principle across the full memory-access path. It fuses two n-gram table lookups, concatenation, active-row discovery, and row registration into a single Triton kernel. The backward pass directly reuses the resulting `row-to-slot` mapping, eliminating duplicated indexing work and preserving sparsity from lookup through gradient update.

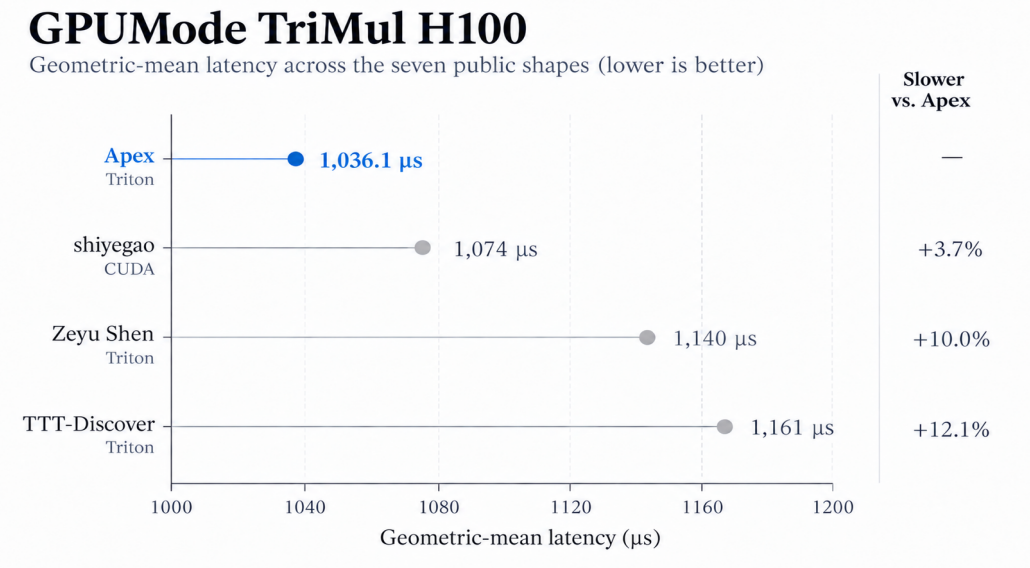
This is a strong systems result because a sparse model no longer pays dense systems costs. The same principle may transfer to other workloads built around large embedding or memory tables whenever each batch touches only a small subset of rows.

Evaluation III: GPUMode TriMul H100

Once a GPU kernel is heavily optimized, the remaining latency is rarely found in the most obvious operation. Bottlenecks move across fusion boundaries, tensor layouts, and memory-access paths, making further progress a test of bottleneck discovery as much as kernel generation.

The GPUMode TriMul task targets the outgoing Triangle Multiplicative Update used in AlphaFold-class protein-structure models. Participants must produce a numerically correct implementation that minimizes geometric-mean latency across seven fixed input shapes on an NVIDIA H100. The task has attracted strong human and automated solutions, including TTT-Discover, published in January 2026 by researchers from Stanford, NVIDIA, Astera Institute, UC San Diego, and Together AI.

Across the seven H100 shapes, our AI system reaches a geometric-mean latency of **1,036.1 µs**. In our same-machine paired evaluation, this improves on the strongest human-designed comparison, stashuk-olek (**1,064.9 µs**), by **2.71%**. The result is also numerically lower than separately reported implementations from shiyegao CUDA (**1,074 µs**), Zeyu Shen Triton (**1,140 µs**), and TTT-Discover (**1,161 µs**), while coming within **0.59%** of K-Search's separately reported 1,030 µs.



Public values are drawn from their respective reports or evaluation records and are not same-machine paired measurements. Our 1,036.1 µs result is measured locally using the official evaluator.

Existing high-performance TriMul implementations already use fusion, specialized matrix multiplication, shape specialization, and layout-aware data movement. Our system initially pushes producer-side fusion further, then detects diminishing and shape-dependent returns as register pressure increases. It consequently shifts the search toward how the producer's output is consumed.

For several workload shapes, our AI system identifies the **BMM→LayerNorm boundary** as the remaining source of latency. The resulting implementation loads BMM outputs along the contiguous physical dimension and performs the required transpose in registers before LayerNorm.

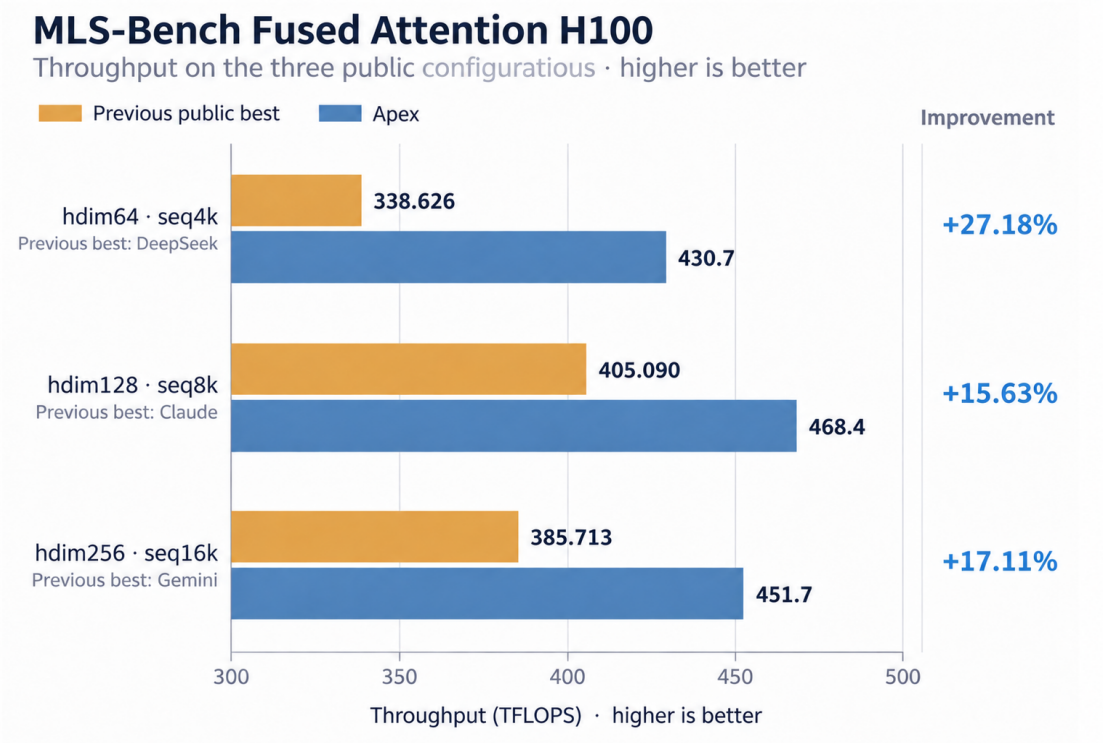
The result is technically significant because our AI system changes the search hypothesis itself: it recognizes when producer-side optimization has saturated, re-localizes the bottleneck across the BMM→LayerNorm boundary, and shifts the optimization target accordingly. Its shape-dependent plans further show that the system can adapt as bottlenecks move across workloads—a capability with broader value than tuning a fixed kernel design.

Evaluation IV: MLS-Bench — Fused Causal Attention Kernel (H100)

Attention is a major compute and memory bottleneck in Transformer training and inference, so its efficiency directly affects model speed and cost. Modern fused-attention kernels are already highly optimized, leaving little obvious headroom; further gains require revisiting the numerical algorithm alongside scheduling and memory movement.

MLS-Bench's Fused Causal Attention task asks AI systems to implement an OpenAI Triton fused self-attention forward pass on an NVIDIA H100, maximize throughput across three causal-attention configurations, and maintain a maximum absolute error below `1e-2`. The benchmark includes results from frontier models such as Claude Opus 4.6, GPT-5.4, and Gemini 3.1 Pro, providing strong baselines from Anthropic, OpenAI, and Google DeepMind.

Across the three fused-attention configurations, our system reaches **430.7, 468.4, and 451.7 TFLOP/s** across head dimensions 64, 128, and 256. These results establish a new best on all three configurations, improving on the previous leading results by **27.18%, 15.63%, and 17.11%**, respectively.



The largest improvement came from simplifying the online softmax path. Standard FlashAttention-style implementations maintain a running maximum and repeatedly rescale the accumulated softmax statistics for numerical stability. Our automated research system found that, on the MLS-Bench workloads, the scaled attention logits stay within a sufficiently narrow range, allowing this repeated rescaling to be removed while still satisfying the benchmark's numerical tolerance.

The resulting **rescale-free softmax** directly accumulates exponentials and weighted values and normalizes once at the end. This improved throughput by **21.0%, 8.5%, and 19.3%** for head dimensions 64, 128, and 256, respectively.

Additional gains came from optimizing shape-specific scheduling and memory movement, including LPT scheduling and TMA. The rescale-free path was the most interesting result because it changed an assumption inside the numerical algorithm itself: a stability mechanism designed for a broader operating range was unnecessary for the workload being evaluated.

More broadly, **this result points to a broader opportunity** that even near the systems frontier, further gains can come from revisiting assumptions that are typically treated as fixed. In this case, Apex's automated research system went beyond tuning an existing kernel's parameters and schedules: it identified a numerical safeguard that could be safely omitted under the benchmark's operating regime and turned that observation into a faster execution path. This is the kind of behavior we are interested in: using search not only to optimize parameters and schedules, but also to challenge the structure of established implementations.

Beyond the Benchmarks

A benchmark captures a result. A recursive AI system carries it forward. Across four benchmarks, our AI research system has produced verified frontier-level improvements. Its process follows a four-stage loop: **discover** a high-value unknown, **verify** candidate solutions, **learn** from the evidence, and **improve** the next cycle. The next step is transfer—turning task-specific advances into reusable knowledge that strengthens the cycles that follow.

Research births the next S-curve. Scientific insight, novel research, and engineered systems carve new pathways — which execution and scale then turn into real-world industries. When one S-curve plateaus, further progress hinges on unlocking the next. We believe the coming technological revolution will be defined by AI acting as a native research engine.

AI is moving from copilot to co-worker. Human judgment and imagination guide what matters; AI expands what can be explored. Together, they can discover new directions, inspire new questions, and build more capable models—creating a reinforcing cycle between machine discovery and human imagination. **Astra shows how far AI has come. What takes it further? The ability to help discover what comes next.**

We are opening these results for others to inspect, reproduce, and extend. The code, experiment records, discovered implementations, and evaluation materials will be released on GitHub.