Agent-AI Assisted Parameter Tuning in Active Electrical Cables for Device Interoperability

Tingting Weng*, Rishik Gupta, Waruna Fernando

Address: 5488 Marvell Ln, Santa Clara, CA 95054, USA
*Email: tweng@marvell.com

Abstract

Active Electrical Cables (AECs) anchor today's scale-out data-center fabrics, with 800 G/1.6 T links on the near horizon to handle the explosive east-west traffic of AI and memory-fabric clusters. In these hyperscale networks, tens of thousands of AEC based links, terminated by heterogeneous switches, accelerators, network interface cards, and co-packaged optics, must establish error free operation on the first attempt and remain stable during hot swap events, airflow transients, and workload driven thermal excursions. Manual or scripted sweeps of Digital Signal Processor (DSP) parameters cannot meet the aggressive operational expenditure and mean time to service requirements at fleet scale. We present a multi-agent AI assisted tuning framework that automates equalization from end to end. The system combines (i) a lightweight prediction engine that estimates link quality in milliseconds, (ii) an on-board adjustment loop that nudges parameters in real time using live bit-error-rate (BER) feedback, (iii) a background explorer that evaluates new setting combinations during low-risk windows, and (iv) a rules engine that enforces vendor limits and rollback safeguards. A rack-level controller arbitrates these inputs, programs only validated values into each cable's DSP core, and continuously refines its models. This agent-based approach delivers production-ready DSP optimization without manual intervention, preserves interoperability across heterogeneous devices, and transforms AECs into self-optimizing, plug-and-play interconnects, thus accelerating current deployments and smoothing the path to upcoming 800 G/1.6 T fabrics.

Introduction

Active Electrical Cables (AECs) are active copper interconnects that integrate a retimer or digital signal processor (DSP) at the cable ends to recover clocks, equalize PAM4 signals, and re-drive clean waveforms, thereby delivering optical-class bandwidth at lower latency and power while extending reach beyond passive direct-attach coppers (DACs). In hyperscale data centers, AECs serve as the short-to-medium-reach workhorses that stitch together switches, network interface cards (NICs) or data processing units (DPUs), and AI accelerators: inside the rack graphics processing unit (GPU)/accelerator to top-of-rack switch and server NIC uplinks, across adjacent racks in mid-of-row layouts, and within dense AI clusters where thousands of lanes must link up quickly and stay stable through thermal and service events. By providing full retiming, adaptive equalization, and standards-compliant link training, AECs maintain margin on challenging channels, support thinner gauges that improve airflow, and reduce operational burden at scale. Marvell's Alaska® A AEC DSP portfolio underpins AEC products for today's 800 G (100 G/lane) deployments and the transition to 1.6 T (200 G/lane),[1] offering robust interoperability with heterogeneous switches and NICs, rich telemetry for fleet observability, and firmware features that enable safe, policy-driven optimization in production environments.

Motivation

In AEC deployments, interoperability with diverse partner switch/NIC ports interacts with cable attributes (e.g. length, gauge, connector stack-ups) and environmental factors (e.g. temperature, airflow, supply voltage and droop), producing a combinatorial explosion of operating conditions and, consequently, a massive parameter space. Beyond sheer scale, the parameter complexity is substantial: on the transmitter (i.e. TX) side, swing and pre/main/post FIR taps and emphasis modes; on the receiver (i.e. RX) side, CTLE indices/gains, DFE taps/thresholds, slicer settings; at the link level, CDR loop bandwidths, adaptation toggles, FEC modes, and training timers; and on the quality side, targets such as BER, SNR and eye margins. As illustrated in Figure 1, traffic between a switch and a NIC via an AEC traverses multiple coupled electrical interfaces in each direction: the switch SerDes to the AEC host-side receiver, the AEC host-side retimed core to the AEC line-side driver, the twinax cable segment itself, and the remote line-side receiver to the NIC host-side SerDes. At each interface, distinct TX controls (for example, FIR pre/main/post taps and swing) and RX controls (e.g. CTLE and DFE) interact with CDR and FEC behavior; link adaptation run bidirectionally, so a small change at one transmitter can alter the optimal receiver settings downstream after retiming and re-drive. The result is multiple, interdependent TX/RX parameter surfaces across the switch, both ends of the AEC, and the NIC, which magnifies the search space and reinforces the need for automated, safety-aware optimization.

Against this combinatorial backdrop, several fleet-scale constraints emerge. Manual sweeps and brute-force grid scans are prohibitively slow, monopolize scarce maintenance windows, and often yield brittle settings that regress outside production conditions. Observability is constrained: measuring truly low BER or rare FEC-tail excursions demands long dwell times, producing sparse, noisy labels and slowing feedback loops. Conventional algorithms struggle with robustness and transferability; preset heuristics and offline optimizers assume stationarity, overfit to specific partners or firmware and thermal states, and rarely incorporate calibrated uncertainty, safe exploration, or transactional rollback. Taken together, these realities motivate an automated, learning-enabled approach capable of reasoning over high-dimensional, mixed discrete–continuous spaces, operating within strict safety and change-rate budgets, and adapting online as devices, channels, and environments evolve.

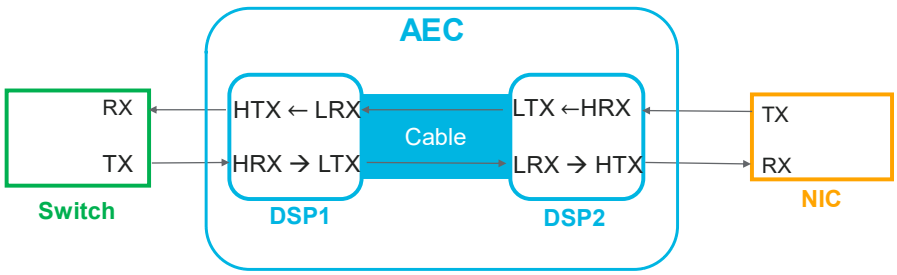

Figure 1. Example of traffic flow at multiple coupled electrical interfaces when connecting AEC with a switch and a NIC.

Multi-Agent AI Framework

To tackle the abovementioned challenges, we implement a multi-agent AI assisted tuning framework that turns AEC links into self-optimizing, plug-and-play interconnects by coordinating four specialized agents under a lightweight orchestration. A compact machine learning (ML) Prediction agent delivers millisecond estimates of link quality; a Live Adaptation agent uses trust-region Safe Bayesian Optimization to make small, safe adjustments in real time; a Background Explorer uses a Genetic Algorithm during low-risk windows to discover better profiles beyond the current operating point; and a RAG-driven Rules Engine synthesizes signed, machine-readable guardrails from curated DSP documentation to enforce bounds, rate limits, and rollback triggers. Orchestrated as a typed state machine with LangGraph and supervised by a rack-level arbiter, the system applies a two-stage commit with watchdogs and dwell timers, records outcomes and reasons for full observability, and updates a profile cache and model or policy bundles through staged rollouts. The following sections detail each agent in turn, covering their inputs and outputs, decision logic, safety constraints, and how they interoperate within the overall orchestration.

(1) ML Prediction Agent

We construct an automatic machine learning prediction agent that will execute both training and inference process. This agent uses Sagemaker AutoML[2] tool to train a compact neural surrogate (e.g., 2–3 hidden layers, 64–128 units) offline using class-balanced sampling and exported as an ONNX bundle[3] for MCU inference. The AutoML tool integrates the end-to-end modeling pipeline that automates the flow from data collection to data cleaning, feature engineering, model training with hyperparameter tuning, followed by model testing and model validation, and finally model deployment to production, as shown in Figure 2.

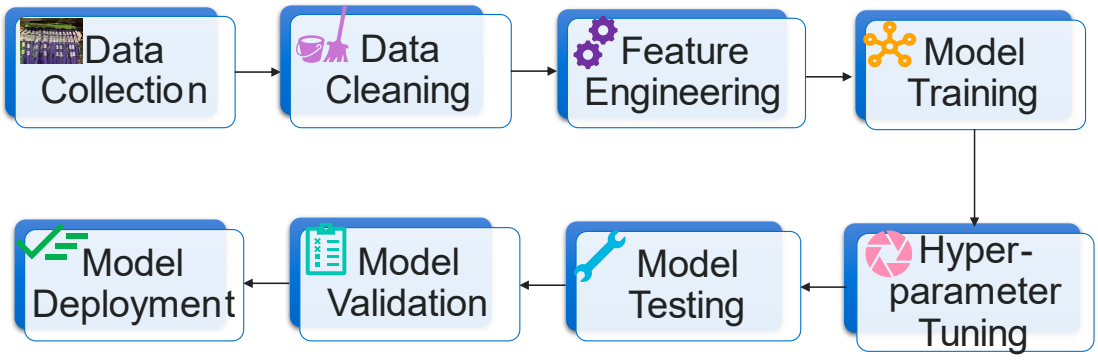

Figure 2. End-to-end machine learning (ML) modeling process.

During the data collection phase, detailed information is ingested including partner identity, cable loss/length/gauge, temperature/voltage, traffic mode, recent BER/FEC statistics, and related DSP parameters such as TX FIR (pre/main/post taps), RX CTLE/DFE settings, and so on. One example

demonstrated below in Figure 3 (a) showing the ML prediction agent auto-train a neutral network model using TX FIR, FFE taps, pga gain, CTLE, DFE etc as input to predict AEC cable channel loss. Figure 3 (b) reflects the high accuracy of the model testing results. The inference layer provides millisecond-class estimates. Based on predicted channel loss, it mapped to pre-stored optimal settings (e.g. TX FIR, CTLE) for this type of cable, applied the optimal settings and then fetch resulting link quality metrics including SNR, BER and FEC statistics.

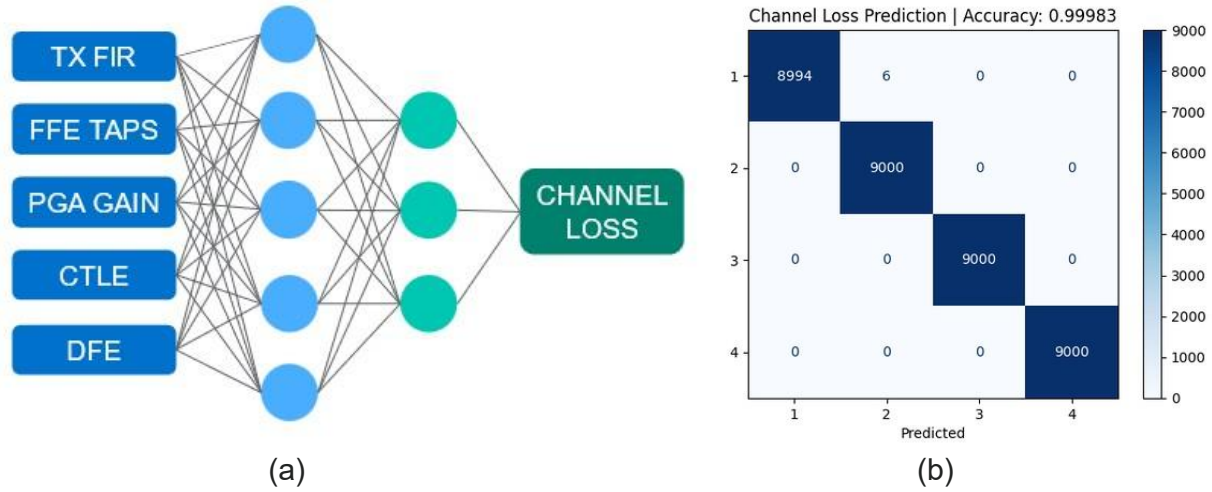

Figure 3. (a) Example of Neutral Network trained by ML prediction agent to predict AEC channel loss; (b) Accuracy result of channel loss prediction model.

(2) Live Adaptation Agent

The live adaptation agent uses a trust-region Safe Bayesian Optimization (Safe BO)[4-7] loop to micro-tune DSP settings online while avoiding unsafe trials with high confidence. It starts from a known-safe seed (for example, the ML prediction agent's recommended profile or the last-known-good setting), builds a small, rate-limited neighborhood around the current configuration (e.g., ±1 step on TX FIR pre/main/post taps and ±1 RX CTLE index), and filters all candidates through the Rules Engine (e.g. bounds, forbidden combinations, and rate limits). Within this neighborhood, the agent treats the prediction model's output as a prior and learns a lightweight local residual model, while a separate safety model estimates tail-risk; candidates are considered only if their conservative lower bound on safety clears a predefined threshold, forming a "safe set." The remaining options are scored with an acquisition that balances predicted improvement, uncertainty (to encourage informative moves), and change size. The top choice is provisionally applied to shadow registers under a watchdog; during a short monitor window the agent evaluates BER, FEC-tail behavior, and SNR proxies, then commits to EEPROM on improvement and stability or rolls back instantly.

(3) Background Explorer Agent

When risk is low (idle periods, redundant paths, soak tests), the explorer searches for better profiles beyond the live agent's tiny trust region using a Genetic Algorithm (GA). GA[8] is a stochastic, population-based optimization method that evolves a set of candidate solutions via repeated cycles of selection (favoring higher-fitness candidates), crossover (recombining "genes" from parents), and mutation (small random perturbations). Because it is gradient-free and naturally explores diverse regions of the search space, GA is effective on noisy, black-box, and highly nonconvex problems, though it can be sample-hungry and benefits from problem-specific encodings and constraint handling. Figure 4 (a) shows the process diagram with GA optimization process. Figure 4 (b) is a screenshot of GA tuning tool that background explorer is using which provides live visualization demonstration of AEC tuning results with 3D contour plot, summary table with optimal results and fitness curve evolved with generation. This GA tuning tool can be also used for quickly tune any untested AEC cables. Compared to manual brute force sweep, this can reduce tuning time from days to just few hours.

For background explorer, a typical GA configuration uses around 100 population, tournament selection (k=3), one-point crossover (p=0.5), and per-gene mutation (p=0.2). Constraint handling combines repair operators with penalty terms derived from the Rules Engine so all candidates remain legal. Evaluations run on shadow links/idle lanes to avoid traffic impact. The best few candidates are handed to the rack-level arbiter for canary trials and possible promotion to production profiles.

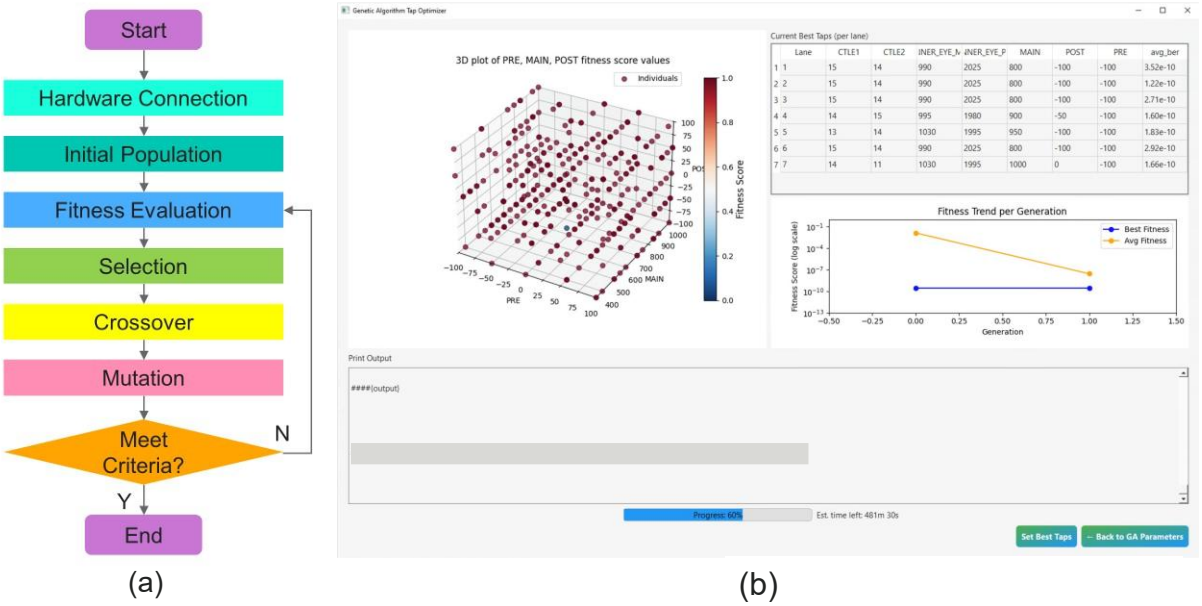

Figure 4 (a) Process diagram using Genetic Algorithm optimization tool; (b) Screenshot of GA tuning tool with live visualization of 3D contour plot, summary table and fitness curve.

(4) Rules Engine Agent

Safety and interoperability are enforced by a rules engine agent based on Retrieval-Augmented Generation (RAG)[9] that synthesizes machine-readable constraints from a curated knowledge base of Marvell DSP rules, standards notes, and field advisories. The agent returns structured policies—per-parameter bounds/quantization, forbidden combinations, rate-of-change limits, dwell timers, and rollback triggers for BER/FEC-tail excursions—plus human-readable explanations and citations for auditability. These policies gate every candidate (live and explorer), shape trust-region steps, and define commit/rollback logic. Together with the arbiter's change budgets and staged rollout, the RAG rules ensure all optimization remains safe, explainable, and compliant while enabling continuous learning from validated outcomes. In practice, this agent is backed by a RAG toolchain with detailed process specified in Figure 5. First, documents from the knowledge base are ingested, split into overlapping chunks, embedded, and stored in a vector database for fast semantic retrieval. At runtime, LangChain[10] orchestrates hybrid retrieval and prompt assembly, and a Large Language Model (LLM) synthesizes the retrieved passages into a structured JSON policy that matches our schema. The service runs headless behind an API, with outputs versioned and signed for the rack controller, and with logs/metrics exported to the existing observability stack for auditability and drift monitoring.

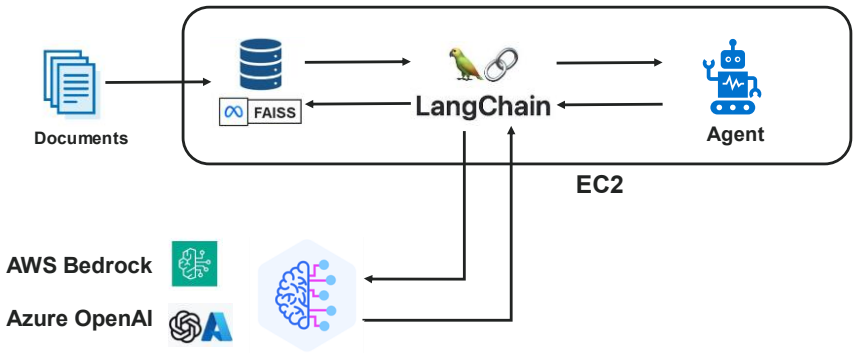

Figure 5. RAG based rule engine tool process using LangChain and LLM

(5) Multi-Agent Orchestration

We orchestrate the abovementioned four agents as a typed, stateful graph with LangGraph[11] running per link and supervised by a rack-level controller. Each cycle, the device loop executes telemetry, ML prediction, RAG rules gating, and then live adaptation, while the controller schedules background GA exploration, arbitrates proposals, and manages model/policy updates. The RAG based Rules Engine emits signed, machine-readable guardrails that every candidate must satisfy. The arbiter enforces change budgets and traffic-safe windows, selects a single action, and applies a two-stage commit: first a provisional apply to shadow registers with a dwell window, then a commit to EEPROM on improvement and stability, or an instant rollback with reason codes. Outcomes feed observability through metrics, traces, and version tags, and update a profile cache keyed by partner, loss, and temperature to accelerate future bring-ups. Model bundles in ONNX

format and policy bundles are versioned and staged through canary, then cohort, then fleet rollouts, ensuring safe, explainable operation while continuously improving priors and validated profiles across the deployment.

Summary

To address the challenges in hyperscale AEC tuning where tens of thousands of 800 G and emerging 1.6 T links must come up cleanly and remain stable through thermal, voltage, and service events, we present a novel tuning solution with a multi-agent AI framework. We integrate an AutoML-trained prediction agent for millisecond link-quality estimates, a live adaptation agent that performs trust-region Safe Bayesian Optimization on live BER feedback, a background explorer that uses a Genetic Algorithm during low-risk windows to discover stronger profiles, and a RAG-driven rules engine that synthesizes signed, machine-readable guardrails from curated DSP documentation. We orchestrate these agents per link with LangGraph and supervise them via a rack-level arbiter that enforces change budgets and a two-stage commit. We cache validated profiles by partner, loss class, and temperature to accelerate future bring-ups, and we version and stage models and policies through canary, cohort, and fleet rollouts to ensure safety and traceability. Compared with manual or scripted sweeps, our approach is safer, more sample-efficient, and markedly faster—reducing tuning cycles from days to hours—while preserving interoperability across heterogeneous switches, NICs, and accelerators, yielding a self-optimizing, plug-and-play AEC fabric ready for 800 G/1.6 T scale.

Reference

[1] IEEE 802.3 Working Group, *IEEE Std 802.3ck™—100 Gb/s, 200 Gb/s, and 400 Gb/s Electrical Interfaces,* 2023.

[2] Amazon Web Services, “Amazon SageMaker Autopilot: Automated machine learning,” service documentation.

[3] ONNX Community, “Open Neural Network Exchange (ONNX) format and ONNX Runtime,” technical documentation.

[4] Y. Sui, A. Gotovos, J. W. Burdick, and A. Krause, “Safe exploration for optimization with Gaussian processes,” in *Proc. ICML,* 2015.

[5] N. Srinivas, A. Krause, S. Kakade, and M. Seeger, “Gaussian process optimization in the bandit setting: No-regret algorithms and experimental design,” in *Proc. ICML*, 2010; extended version in *IEEE Trans. Info. Theory,* 2012.

[6] D. Eriksson, M. Pearce, J. Gardner, R. Turner, and M. Poloczek, “Scalable global optimization via local Bayesian optimization (TuRBO),” in *Proc. NeurIPS,* 2019.

[7] M. A. Gelbart, J. Snoek, and R. P. Adams, “Bayesian optimization with unknown constraints,” in *Proc. UAI*, 2014.

[8] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, MA: Addison-Wesley, 1989.

[9] P. Lewis, E. Perez, A. Piktus, *et al.*, “Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks,” *NeurIPS*, 2020.

[10] LangChain, “LangChain: Building applications with LLMs,” framework documentation.

[11] LangGraph, “LangGraph: Typed, stateful agent workflows,” framework documentation.