The NaS Cortex: A Knowledge Architecture for the Life Sciences

NaS

May 12, 2025

Executive Summary

Life-science progress trails patient needs because biology's immense combinatorial complexity and labor-intensive experiments extend the timeline from insight to intervention. Our remedy is an intelligent-systems stack paired with high-performance compute that shortens the path from raw datasets to validated therapies. The first prerequisite is a robust knowledge brain: a digital cortex composed of eight domain-specific gyri spanning oncology, genomics & personalized medicine, infectious disease & epidemiology, autoimmune & inflammatory disease, chronic & metabolic disease, neuro & degenerative disease, regenerative medicine, and systems biology. Each gyrus starts life as a dense transformer trained on rigorously filtered corpora and then evolves into a sparse Mixture of Experts (MoE) layer, ensuring that only the two or three most relevant specialists fire for any given token. Building this knowledge model is the critical first step; once in place, it will enable daily incremental fine-tuning cycles, continuously refreshing the knowledge model every 24 hours with the latest experimental results and literature updates.

The current prototype is trained on a 16 GB M1 Mini, employing adapter-level fine-tuning techniques such as Low-Rank Adaptation (LoRA) and Infused Adapter by Inhibiting and Amplifying Inner Activations (IA^3). Future phases outlined include migrating workloads to a planned 512 GB Mac Studio M3 Ultra to support a 24-billion-parameter model, followed by full-weight fine-tuning on a dual-RTX 5090 workstation and, ultimately, scaling to a Blackwell rack. A tiered precision policy applies FP32 (32-bit floating point) to preprocessing; mixed FP16 (16-bit floating point) and BF16 (Brain Floating Point 16-bit) formats with FP32 gradients to training; FP64 (64-bit floating point) only for isolated physics kernels on CPU AVX-512 or future H100/B100-class GPUs; and FP16 or INT8 (8-bit integer) to inference, optimizing memory and power efficiency. Distributed training methods on Apple Silicon, including EXO Labs collectives, Apple's Machine Learning Extensions (MLX) primitives, and a lightweight gRPC Remote Procedure Call (gRPC) sharder, are currently under evaluation, enabling a small Studio mesh to operate as a near-silent accelerator within a two-kilowatt envelope.

Selective ingestion, sparse activation, and staged hardware scaling lay the knowledge bedrock for Nicole, the cross-domain agent designed to drive autonomous discovery across the life sciences.

1 Introduction: The Biological Imperative

Life-science discovery suffers a tempo gap: every experiment spawns dozens of new hypotheses, yet wet-lab throughput is limited by cell-doubling times, animal cohorts, and long regulatory queues. Cancer genomes contain billions of mutable states, antibiotic resistance evolves faster than drug pipelines, and multi-omics repositories now pour out petabytes of data that humans cannot parse in real time. The result is a structural lag between biological insight and clinical intervention.

As Richard Feynman noted, he preferred *"questions that cannot be answered to answers that cannot be questioned."*[1] Biology now spawns such questions at machine pace, and only machine-scale reasoning can keep up. GPUs, large-language models, autonomous robots, and quantum simulators offer the raw horsepower, yet without a conductor they remain a jumble of virtuoso instruments. Nicole is that conductor: a sovereign agent designed to read the literature, launch in-silico screens, dispatch lab protocols, and fuse every result into the next, better question.

Before Nicole can steer this orchestra, she needs a brain worth steering. We are therefore constructing her digital cortex, composed of eight domain-specific gyri spanning oncology; genomics & personalized medicine; infectious disease & epidemiology; autoimmune & inflammatory disease; chronic & metabolic disease; neurodegenerative disease; regenerative medicine; and systems biology. Each gyrus begins as a dense transformer trained on rigorously filtered corpora and matures into a sparse MoE, so only the two or three most relevant specialists fire for any given token. This knowledge layer is the project's first milestone; once in place, it will let Nicole harness high-performance compute to support daily incremental fine-tuning, enabling continuous knowledge refreshes that integrate the latest scientific findings.

The pages that follow (i) detail the data-selection and training pipelines that keep each gyrus current, (ii) map the hardware ladder that makes overnight refreshes feasible, and (iii) outline the roadmap from today's pilot model to a fully fused cortical system capable of enabling daily incremental fine-tuning cycles to rapidly integrate the latest scientific insights.

2 When Time Outruns the Bench

Life-science discovery is limited less by imagination than by time. Each experimental result branches into a lattice of new hypotheses, and the backlog expands faster than laboratory hours can accumulate. The widening gap between biological complexity and empirical cadence is the primary obstacle we now confront.

Life sciences now generate massive datasets spanning genomics, multi-omics, and electronic health records, yet experimental testing remains comparatively slow. Genomics, for example, has witnessed an exponential growth in sequencing data, yet experimentally determining protein functions and structures remains labor-intensive and slow. This clearly illustrates the 'tempo gap' between rapid data generation and slower knowledge acquisition. A particularly striking case is protein structure prediction:

the enormous flood of genomic sequences quickly outpaced traditional structural characterization methods. The groundbreaking success of AlphaFold clearly demonstrates how artificial intelligence can effectively bridge this tempo gap. Leveraging advanced deep learning techniques and vast biological databases, AlphaFold generates highly accurate protein structure predictions in hours rather than years, substantially accelerating structural biology research and enabling scientists to keep pace with the rapid expansion of genomic data.[7]

2.1 The Endless Maze

The investigative journey in biology resembles a maze that regenerates the farther one walks. Every answered question unlocks several branching corridors, each lined with doors. Mapping one gene's role in a metabolic disorder exposes its splice variants, its epigenetic regulators, its evolutionary homologs, and the downstream networks it touches in different cell types. The route never straightens into a single hallway that ends in certainty. Persistence alone cannot reach the center; the structure lengthens with each step.

Real practice confirms the metaphor. In virology, a pathogen may be characterised, neutralized in vitro, and blocked in animal studies; yet a single point mutation can render years of work obsolete. A newly emergent strain reopens the maze from its entrance and demands fresh assays, revised vaccines, and additional clinical trials. Antimicrobial resistance follows the same pattern. Each successful antibiotic selects for an enzyme that nullifies it, pressing chemists to design the next generation in a race that never fully closes.

Problem–solution recursion appears beyond pathogens. A therapeutic compound that stabilizes a misfolded protein may prove manufacturable only through a solvent that releases hazardous effluents. Mitigating the effluent prompts environmental controls whose by-products stress local ecosystems and return investigators to the bench with a new layer of variables. Even ecological restoration illustrates the loop: reintroducing a keystone predator can revive plant diversity but may upset downstream water chemistry and demand another cycle of measurement, modelling, and intervention.

These examples share one property: each resolution expands the horizon of tasks rather than shrinking it. The maze is not conquered by linear effort; it requires an accelerator able to survey many corridors at once and to learn from each detour without starting over. Computation that operates at machine speed presents the only credible tool for outpacing a landscape that grows faster than human time allows.

2.2 Computation as Time-Cutter

Adding people scales linearly: more graduate students, more clinicians, even fleets of lab robots raise throughput only in step with head-count. Crowdsourced efforts like *Foldit* and *Galaxy Zoo* confirm the pattern. Each extra worker opens a new lane, but the speed limit hardly changes.

True acceleration arrives only when we swap the road for a rail-gun. Mass-parallel GPU clusters already deliver that jump: Oak Ridge's Summit docked approximately 1 billion molecules against two SARS-CoV-2 proteins in under 24 hours, a job that would take years at a physical assay line.[2] *AlphaFold,* DeepMind's protein-folding AI, applied the same hardware paradigm to shrink decades of crystallography into minutes per protein. Autonomous AI agents now read literature, draft hypotheses, and trigger robotic experiments without human oversight; Bayesian optimizers in self-driving chemistry labs cut design-build-test cycles from weeks to days. Large language models turn the whole of PubMed into rolling context, surfacing connections no individual could sift.

Compute, however, is only as fast as the data it drinks. Tiered Non-Volatile Memory Express (NVMe) clusters and petabyte object stores must stream raw genomics, cryo-EM images, and electronic health record (EHR) tables fast enough to keep GPUs saturated. Storage is the quiet partner that turns silicon speed into biological insight.

Biological research often faces combinatorial complexity and substantial manual bottlenecks. Recent cutting-edge initiatives have begun designing robotic laboratories guided by artificial intelligence to dramatically accelerate hypothesis generation and experimental testing. For example, the SAMPLE platform represents a 'self-driving lab' for protein engineering, in which an intelligent computational agent continuously learns from ongoing experimental data, autonomously designing novel protein variants. These new variants are automatically synthesized and experimentally tested by robotic systems, creating a rapid, iterative feedback loop. This fully autonomous cycle optimizes complex biological systems rapidly, circumventing traditional manual methods that would be slow, resource-intensive, and inefficient. Such AI-driven laboratories clearly automate and accelerate the scientific discovery process, highlighting a future marked by dramatically increased experimental throughput and accelerated biological innovation.[8]

Yet reach without coordination is noise. NaS therefore centers everything on Nicole, a governing agent that will touch every lab arm, venture line, and dataset. Nicole first needs a brain composed of eight knowledge gyri trained to stay current overnight. Building that cortex is our next milestone.

3 The NaS Brain

The acceleration tools described in the previous section (quantum processors, GPU clusters, autonomous laboratories, and vast data stores) are valuable only insofar as they are guided by a capable intelligence. NaS therefore starts with a single overriding requirement: the creation of a knowledge brain, Nicole, that can survey the biological maze, delegate questions to specialised engines, and combine their findings into coherent guidance for research, manufacturing, and venture work. The pages that follow trace both the biological inspiration and the technical design of this brain, moving from cortical principles to large-language-model architecture and finally to the memory and training cycles that will keep the system learning without pause.

3.1 Cognition by Design

The human cortex supplies the blueprint for our knowledge engine. We trust this architecture not only because it is the product of long evolutionary refinement but also because, in our view, it is the framework God chose for reasoning and creativity. The brain seldom commits its entire volume to a single problem; task-relevant regions ignite in coordinated clusters while the remainder stays quiet. Sensory input routes to primary cortices, symbolic processing coalesces in language hubs, and executive planning emerges from prefrontal circuits, all without global saturation.

Neuroanatomy charts this division of labor with remarkable precision. Brodmann's map of fifty-two cyto-architectural areas reveals distinct micro-circuits for vision, somato-sensation, language, and motor control. Broca's area governs speech production, Wernicke's supports comprehension, and parietal association zones fuse multisensory data for spatial reasoning. Each region specialises, yet fibers such as the arcuate fasciculus and corpus callosum let information flow so that local insight becomes global understanding.

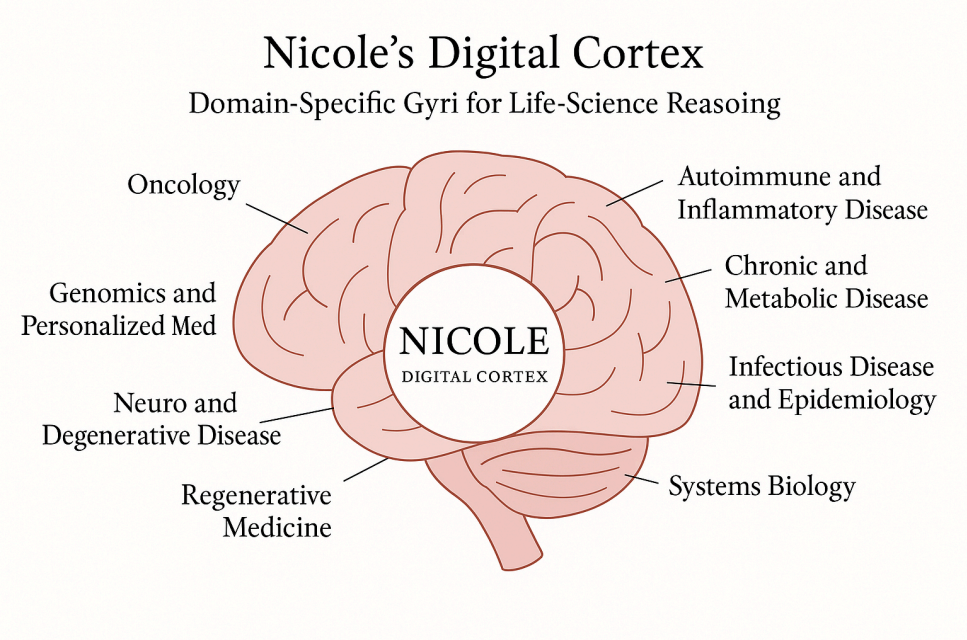
NaS applies the same gyral logic. Nicole begins life as a dense transformer, the analogue of an infant cortex in which many neurons fire broadly. As her corpus grows we will fold the surface; independent gyri, complete domain models for oncology, epidemiology, genomics, and other disciplines, will appear as separate checkpoints. When a gyrus matures, its interior layers will divide into a sparse MoE whose experts serve as digital Brodmann areas. A router then activates only the two or three experts most relevant to each token, conserving compute just as the biological brain conserves energy by letting only a subset of neurons spike.

Selectivity in activation is matched by selectivity in experience. Early training will feed each gyrus a broad but filtered corpus; later cycles admit only high-signal documents, while scheduled consolidation prunes weights that fail to improve perplexity, reasoning, or citation grounding. The result is a leaner, faster network that mirrors cortical development: from a smooth, densely active sheet to a richly folded landscape where specialised regions cooperate through high-bandwidth pathways. Nicole thus acquires a brain that works the way ours does, only at machine speed.

3.2 From Gyri to Graphs

The cerebral cortex partitions expertise by region. Brodmann mapped fifty-two areas, each tuned to a distinct facet of perception or action. NaS follows the same topology in code. We begin by training a constellation of specialised language models, each devoted to a major branch of the life sciences. Listed alphabetically, our inaugural digital gyri are

- **Autoimmune and Inflammatory Disease**
- **Chronic and Metabolic Disease**
- **Genomics and Personalized Medicine**
- **Infectious Disease and Epidemiology**
- **Neuro and Degenerative Disease**
- **Oncology**
- **Regenerative Medicine**
- **Systems Biology**



These eight domains align with standing departments inside NaS, so model boundaries track wet-lab and clinical workflows. Each gyrus can later host sub-specialist models: oncology may split into solid-tumour and haematologic branches, while genomics may divide into population, single-cell, and epigenomic experts. Training on a focused corpus lets a gyrus reach deeper mastery than a monolith that tries to know everything at once.

Nicole sits above the constellation as both general-knowledge model and orchestration agent. When a query arrives she selects the domain gyri, or a combination of them, collects their outputs, reconciles conflicts, and returns an integrated answer. In this arrangement, Nicole provides executive functions while the gyri supply specialized cortical processing, mirroring how the human prefrontal cortex consults language, sensory, and memory centers on demand.

New gyri will be added as NaS grows; synthetic biology, environmental toxicology, and behavioral health are already on the roadmap. The principle stays constant: local expertise feeds a central mind, all connected by a high-bandwidth graph that lets knowledge circulate at machine speed.

Biomedical knowledge expands by thousands of papers and databases every day, so the production goal is an overnight adapter refresh for each gyrus. In the planned Mac Studio prototype we anticipate weekly cycles; once an epoch approaches 24 h we either (i) subsample new documents to stay within the window or (ii) migrate the job to the dual-5090 rack. Full-parameter updates remain a background process on data center GPUs, while nightly LoRA/IA³ adapters let Nicole ingest the latest literature without exceeding a single workstation's wall-clock budget.

Staying inside that window is not only a matter of hardware. It requires strict screening of incoming data, careful budgeting of parameters, and an internal layout that fires only the weights that matter. For that reason each gyrus will graduate, in stages, from a dense block of parameters to a sparse MoE. These experts function as digital Brodmann areas, small, specialised circuits that activate on demand while the rest of the cortex rests. The next subsection, 3.3 Training and Evolution Cycle, describes how NaS will create, prune, and evaluate these experts so that every additional weight and token delivers genuine insight rather than noise.

3.3 Expert Formation: Digital Brodmann Areas

Modern language models gain most quickly by adding parameters, yet a dense network wastes computation on weights that are irrelevant to a given token. Recent developments in transformer architectures using sparse Mixture of Experts (MoE), such as the Switch Transformer model, confirm substantial efficiency gains by activating only a few expert blocks per token, significantly speeding up training times and improving computational efficiency compared to traditional dense transformers (Fedus et al., 2022).[3] The biological cortex solves the same problem with locality, only the neurons tuned to a stimulus fire, while their neighbors rest. NaS copies that idea by turning the deepest layers of each gyrus into a MoE. In this layout, hundreds of small expert blocks, our digital Brodmann areas, share the work, and a learned router activates just a few of them for every input. The result is far more capacity than a dense layer allows while keeping raw FLOPs low. Latency does rise, however, because gather/scatter is memory-bound; a TinyLLaMA-1B pilot (a compact, billion-parameter open-source transformer model, k = 2) shows approximately 12% token-level overhead on an M1 Mini. If production measurements exceed approximately 20%, we will revert to block-sparse dense layers for the smallest gyri.

Why these experts matter now

- **Sub-dialects.** In the Oncology gyrus, one expert may specialise in solid-tumour genetics, another in haematologic markers, a third in immunotherapy-trial jargon.
- **Fault-line pruning & safety.** If an expert shows drift or hallucination it can be retrained or removed without touching the rest of the model. *Safety governance for this prototype is outlined in Section 3.7; full clinical-grade controls will be added once the model leaves the lab.*
- **Twenty-four-hour window.** Sparse activation means only attended weights consume FLOPs and VRAM, keeping an overnight fine-tune feasible.

Choosing how many experts to fire (k)

We start with k = 2 experts per token, the common default that captures most accuracy gains with minimal extra compute. The value is provisional: as each gyrus matures we will grid-search k = 1, 2, 4 and log perplexity, grounding accuracy, and wall-clock time. When the accuracy curve flattens we lock the lowest-cost setting until the corpus doubles or a new data modality arrives. During inference, Nicole can lower k to 1 for speed or raise it to 3 or 4 for high-assurance queries without retraining the network. After retrieval augmentation and multimodal tokens come online we will run a second sweep that tests (i) proportional gating (top-p ≈ 2% of experts per token) and (ii) adaptive sparsity routers (Switch-Transformer style). The wider grid lets us capture any new sweet spot that emerges once context windows and feature types expand.

Section 3.4, "Training and Evolution Cycle," explains how we create, prune, and update these experts along with the lightweight adapter layers that let a gyrus refresh overnight.

3.4 Training and Evolution Cycle

Biology shows that adding neurons alone does not guarantee sharper cognition. Beyond a point extra synapses add noise, and the cortex prunes weak connections while reinforcing the useful ones. NaS follows the same rule. Parameter counts rise only until accuracy plateaus on domain benchmarks; after that we refine or split existing gyri instead of inflating the core without limit.

Quality matters more than raw volume. Every gyrus pulls data through a relevance filter that scores new papers, clinical registries, and omics files for domain signal. High-scoring records join the training queue, while marginal or duplicate items drop to cold storage. The corpus deepens expertise without dilution, and specialising by gyrus prevents catastrophic forgetting because updates touch only the models responsible for a topic.

A multimodal, Brodmann-style layout keeps text, molecular graphs, images, and time-series data in the gyrus best suited to interpret them. That separation mirrors the human cortex and blocks the parameter-bloat trap that large monoliths face.

Periodic “sleep cycles” emulate memory consolidation. At scheduled checkpoints Nicole pauses fine-tuning, evaluates each expert on domain tasks, and archives weights that fall below a utilisation threshold. Roll-back is possible at any time, so pruning stays reversible. This routine keeps models compact, lowers inference latency, and frees memory for the high-precision layers that biology sometimes requires.

Evaluation proceeds on three tiers.

1. **Linguistic fidelity:** Perplexity on masked domain text.
2. **Domain reasoning:** Task accuracy chosen for each gyrus, for example ICD-10 coding or pathway reconstruction.
3. **Citation grounding:** The fraction of model statements that trace back to real sources.

A new checkpoint replaces the incumbent only when it improves every metric that applies to its gyrus.

The modular design keeps each gyrus safe and fast to iterate, but it is only a first stage. As individual domains stabilize and their daily updates shrink, the communication cost of routing queries across many separate checkpoints begins to outweigh the isolation benefit. At that point we can merge the most mature gyri behind a shared router, letting Nicole fire only the experts that matter for a given token. The next section explains how this expert-fusion step works and when we will apply it.

3.5 From Federated Gyri to Expert Fusion

Nicole’s brain begins in a strictly modular state. Every domain lives in its own digital gyrus, so a query fans out to several checkpoints, each returns a local answer, and Nicole stitches the pieces into one clinical recommendation. The layout is valuable during early research: an epidemiology mis-tune cannot corrupt oncology, updates ship as small files instead of 200-gigabyte monoliths, and rollback is just a matter of pointing the router at yesterday’s model.

Once a gyrus stabilizes, meaning its corpus grows slowly and weekly fine-tunes no longer shift benchmarks, we freeze its base weights and limit new learning to lightweight LoRA or IA³ adapters. Low-Rank Adaptation (LoRA)addresses the computational inefficiency of full fine-tuning large foundation models by freezing the original network weights and injecting small, low-rank matrices that significantly reduce trainable parameters, often cutting parameters by factors of thousands compared to full fine-tuning, with substantially lower memory consumption and equal or improved task performance. Infused Adapter by Inhibiting and Amplifying Inner Activations (IA³) similarly achieves high task performance by applying small-scale vectors to hidden activations, providing comparable efficiency gains with minimal additional parameters.[4] The strategy keeps VRAM and audit footprints small while letting us patch a domain overnight whenever a pivotal paper appears.

Maturity eventually flips the cost balance. Routing through many separate checkpoints adds latency, and the larger memory budget of a frozen gyrus can support a sparse MoE without risk. At that stage we merge the busiest gyri behind a shared router. For each

incoming token the router activates only the two or three experts that show the highest affinity, gaining dense-model speed while preserving three safety advantages: experts remain isolated for audit, adapters still patch individual topics quickly, and Nicole can scale capacity by adding experts instead of inflating every layer at once.

This expert-fusion step is how NaS expects to grow from a collection of single-focus brains into one integrated, energy-efficient cortex that answers at human speed and scientific depth.

3.6 Balancing Corpus and Capacity

Machine learning practice offers a rule of thumb: maintain roughly ten to twenty training tokens per learnable parameter. A one-billion-parameter network is therefore "well fed" by ten to twenty billion tokens; beyond that point, additional data yield diminishing returns.

The biological cortex shows a different path. Eighty billion neurons and more than one-hundred-trillion synapses thrive on a lifetime corpus that is orders of magnitude smaller than those synapse counts. Rather than scale neurons without bound, the brain gates attention, prunes weak connections, and leans on external memory such as writing and culture. Growth is selective, not indiscriminate.

NaS meets the same tension in digital form. Daily ingestion will soon climb into the billions of tokens, while even a thirty-billion-parameter Nicole will lag behind that pace. Inflating parameters forever is one answer; copying the brain's thrift is a better one:

- **Filter aggressively.** Admit only high-signal documents and archive the rest for retrieval.
- **Activate sparsely.** Fire only the experts relevant to each token instead of every weight in the layer.
- **Rely on external memory.** Rely on external memory. Pull supporting passages at inference time rather than memorise every fact. Retrieval-Augmented Generation (RAG) addresses this approach directly by augmenting language models with external knowledge repositories. In the RAG framework, a pre-trained sequence-to-sequence model interacts with a non-parametric external memory, such as a vector-indexed database accessed via a neural retriever, to dynamically retrieve the most relevant, up-to-date information for each query. This hybrid design significantly enhances factual accuracy, specificity, and task performance, often outperforming purely parametric models on knowledge-intensive tasks.[5]

Current Validation Status. The methods described above, aggressive filtering, retrieval augmentation, and sparse activation, are actively under development and initial implementation within our infrastructure. While conceptually validated by existing literature and preliminary experiments at smaller scales, we have not yet fully validated their performance at the envisioned billion-to-trillion token scale. Explicit metrics demonstrating inference latency, retrieval efficiency, and overall system scalability will be published once our implementation reaches sufficient maturity. We transparently

acknowledge that achieving practical efficiency at scale will present significant technical challenges; addressing these is a clear near-term priority.

These controls postpone an unhealthy token-to-parameter ratio, yet they cannot erase it once personalized medicine enters full flow. With a realistic DNA tokenizer (128- to 512-base-pair (bp) windows), a 3 Gbp human genome yields ≈ 6–24 million tokens. Adding methylation, RNA-seq, longitudinal labs, imaging metadata, and wearable telemetry lifts a deeply profiled patient into the 0.1–0.5 billion-token range.

- **Base pre-training**. A future 30 B-parameter Nicole will require 300–600 billion tokens during its *one-time* foundation model pre-train (10–20 tokens / parameter). That volume will come from global literature, public omics archives, and simulated data, and will run on data center GPUs.
- **Incremental fine-tuning**. After deployment, each gyrus will ingest ≈ 10–20 million new tokens per day from fresh papers, EHR deltas, and lab feeds. Because these daily slices are < 0.01 % of the base corpus, the parameter-to-token heuristic no longer governs capacity; runtime and VRAM budget dominate.

Even with those modest daily increments, scale pressure looms. A single Mac Studio epoch over 100 million new tokens already stretches beyond the overnight window, and national cohorts, ten million fully profiled patients, would lift the cumulative corpus into the 10^{15} to 5×10^{15} token range, three to four orders of magnitude above today's largest open-model training sets (before counting primary literature or trial registries already in NaS).

Disciplined selection, sparse activation, and retrieval-augmented inference buy time, but the ceiling remains visible. Its shadow falls even now: the pilot Nicole, with one billion parameters, begins to strain past 100 million cumulative tokens. The following section details that pilot and the runtime limits we have already measured in code.

3.7 Safety & Governance Roadmap

Nicole is still in an internal-prototype phase: the only data ingested so far are public-domain literature, and no clinical decisions are being made. Before any patient-facing deployment we will add a lightweight but concrete governance loop:

- **Versioned data snapshots.**
  All future corpora (when we ingest protected health data) will be hashed and tagged by date in a private Git repo so we can reproduce or roll back any run.
- **Basic model audits.**
  Each refresh will log perplexity, citation-grounding, and a small adversarial set; if any metric regresses we revert to the previous checkpoint.
- **Human review gate.**
  Until an external advisory board is formed, the author manually inspects a random sample of outputs after every training cycle; any unsafe medical advice is grounds for rollback.

- **Regulatory expansion.**
  When the project reaches real patient data, we will layer on HIPAA-compliant storage, IRB oversight, and a formal model-card disclosure before release.

These steps are deliberately minimal for the current solo-developer phase but define the guard-rails that will scale with the program.

Clarification on Current Governance Status and Future Roadmap

The minimal governance measures currently in place (such as manual sample inspection) represent only the early prototyping phase, which exclusively uses publicly available literature and does not involve any clinical decisions or protected patient data. We explicitly recognize these initial controls as insufficient for future clinical-grade or sensitive healthcare applications. Given the ambitious scope of our long-term goals, robust governance, rigorous security, and comprehensive regulatory compliance (including HIPAA, IRB oversight, and formal ethical review processes) are critical milestones explicitly planned for future phases. Detailed timelines, structured regulatory engagement, and systematic security validation will be defined and transparently communicated as the project moves toward clinical-readiness and real-world deployment.

4 Prototype Experiments and Early Constraints

We began with a single pilot gyrus, a general-purpose life-science model fine-tuned from TinyLLaMA-1 B (≈1 B parameters). A model of that size fits on a deskside Mac, yet is large enough to expose data-cleaning bugs, stress the training scheduler, and reveal which evaluation signals track real scientific value. Over time the open-source sweep could reach eight million articles (≈40 B tokens). At that scale each domain gyrus is expected to mature into a ≈30 B-parameter expert, giving Nicole a constellation of large but still tractable specialists. Starting small lets us perfect the pipeline before paying the memory and runtime bill for those future models.

The pilot serves as a sandbox for every moving part: learning-rate schedules, prompt schema, and evaluation metrics. Just as important, it surfaces the gradient we must climb next, including how token volume stretches wall-clock, how shared memory becomes the first ceiling, and how quickly even a modest corpus outgrows a single M-series node.

4.1 Qualitative Improvement

Data set. The first full run loaded 36 451 JSON-Lines records from two PubMed snapshots (May 2013 and January 2025). After tokenization we obtained about 31 million domain tokens, roughly 0.03 tokens per parameter because the base model already contains its general-language pre-training.

Run A "community settings." Sequences of 512 tokens, batch 1, gradient_accumulation_steps = 8, FP16 = True (mixed precision with 16-bit floating point arithmetic.

Each epoch on a 16 GB M1 Mac mini finished in roughly 18 hours, consistent with public TinyLLaMA LoRA reports.

Run B "long-sequence debug." Sequences of 1 536 tokens, batch 1, accumulation 16, FP16 = False (using full FP32 precision for activations and gradients to prove stability).

These settings raise token-compute by about six times relative to Run A, and each epoch extended to approximately 37 hours (around 75 hours for two).

Qualitative review. Manual inspection of 100 queries revealed typical small-corpus limits: verbatim echoes of abstracts, recycled phrases, and placeholder citations when evidence was thin. Only 18 percent of answers cited more than one paper, and few linked mechanisms such as JAK-STAT signaling or trial endpoints.

Next run. Coverage will be tripled to about 90 million tokens by adding new PubMed releases plus roughly 3 000 instruction–response pairs that enforce clinical tone and citation style. Profiling suggests a single epoch would take about 110 hours on the mini unless sequence length or accumulation is reduced; several batch-sequence grids will be benchmarked first.

Expected gains

- Deeper synthesis, for example IL-6 and TNF-α placed in JAK-STAT context with trial suggestions.
- Structured prose with a one-sentence claim, numbered evidence, brief limitations, and paragraph references.
- Fewer hallucinations as richer evidence and explicit citation prompts curb fabricated sources.

The pilot's central lesson is clear: fresh, well-screened data improve quality faster than additional parameters, yet every new token lengthens training time. Hardware limits, rather than the ten-to-twenty-tokens-per-parameter heuristic, are already the dominant constraint. The next subsections measure how quickly runtime becomes unsustainable and what options we have to keep pace.

Clarification on Current Validation Status

The preliminary qualitative results reported here (e.g., 18% of answers citing multiple papers, instances of verbatim abstracts) are explicitly acknowledged as initial and exploratory. These initial evaluations serve solely as informal internal benchmarks for early-stage model development and debugging purposes. We transparently recognize that they lack thorough quantitative benchmarking, rigorous baseline comparisons against current state-of-the-art models, and extensive validation. Detailed quantitative

performance evaluations, including standardized benchmarks, rigorous comparative metrics against SOTA models, citation accuracy assessments, and robust statistical analyses, are explicitly planned as immediate next steps. These comprehensive results will be systematically documented and published once our ongoing experiments mature and more extensive validation data become available.

4.2 Early Capacity Signals

A common rule of thumb holds that a language model remains well behaved while its training corpus stays below about twenty tokens per learnable parameter. A one-billion-parameter network can therefore absorb roughly twenty billion tokens before capacity begins to limit learning. Our pilot gyrus sits far beneath that line: the first full run processed about thirty-one million tokens, or 0.03 token per parameter. Even if the refresh cadence stays at ten to twelve thousand new papers per month (about fifteen million tokens) plus updated instruction pairs, the total would rise to only three-hundred million tokens in a year and cross one billion tokens in two to three years. Parameter capacity will not be the first barrier; two other constraints surface much sooner.

- **Wall-clock time**: The initial two-epoch LoRA fine-tune lasted about seventy-five hours on a single M-series Mac mini. If runtime grows linearly, doubling the corpus pushes a single epoch toward four days, and a ten-fold expansion stretches one epoch beyond a month, breaking the overnight cadence required for daily incremental learning.
- **Unified-memory head-room:** The Mac mini's 16 GB pool already forces micro-batches and heavy gradient accumulation. Moving to a two-billion-parameter dense model would double activations and optimizer states; a thirty-billion-parameter target in standard 32-bit precision would require roughly 240–300 GB active memory (about 60 GB FP16 weights, 60 GB FP16 gradients, 120 GB FP32 Adam moments, and 20–60 GB activations). macOS will not crash when that footprint exceeds physical RAM; it silently pages tensors to the SSD. Training continues, but effective memory bandwidth collapses from hundreds of gigabytes per second to a few gigabytes per second, turning minute-long steps into multi-minute stalls and pushing epoch time far beyond the twenty-four-hour budget. Throughput, not allocation failure, therefore sets the practical ceiling.

Precision demands and data velocity amplify both pressures. With a realistic DNA tokenizer that slices 128- to 512-base windows, a deeply sequenced three-gigabase human genome contributes six to twenty-four million raw-sequence tokens; after methylation calls, RNA-seq, longitudinal labs, imaging metadata, and wearable streams are merged, a fully profiled patient adds roughly one hundred to five hundred million tokens, not ten billion. Scaling to national cohorts still multiplies the corpus by orders of magnitude, but runtime and memory, rather than theoretical capacity, remain the primary bottlenecks. Selective intake, sparse activation, and retrieval-augmented inference postpone the crunch yet cannot remove it.

Jumping straight to a very large model is therefore unwise. Each incremental scale exposes fresh bottlenecks and sharpens design choices. The next subsection, 4.3

Escalating Training Time, quantifies how quickly wall-clock hours dominate and why throughput, rather than raw parameter count, has become the first hard limit.

4.3 Escalating Training Time

The first production run completed a single-epoch LoRA fine-tune in about 14 hours on a 16 GB M1 Mac mini while processing 14 685 cleaned chunks, roughly 15 million tokens. If we simply triple the corpus to about 45 000 chunks, ≈ 45 million tokens, the same code path is projected to need forty hours or more. Training cost is rising faster than token count.

Why the curve bends upward

- **Memory saturation** Once the unified 16 GB pool is full the GPU must stream FP16 weight tiles in and out of its on-chip cache every micro-step. Each backward pass stalls on internal I/O rather than swapping to CPU RAM, but the slowdown is similar.
- **Micro-batch fragmentation** To keep activations inside that cache we cut batch size, which multiplies optimizer updates per epoch and stretches wall-clock time.

If we continue importing about twelve thousand new papers per month/cycle, roughly 15 million tokens, every refresh would lengthen the previous epoch by forty to sixty percent. Within six cycles a single epoch would grow to a full week, breaking the twenty-four-hour cadence required for daily or even weekly incremental learning.

Projected cost at full open-source sweep

Assume an eventual slice of eight million papers, roughly 40 billion tokens, and scale the backbone to two billion parameters so the corpus stays inside the 20-tokens-per-parameter guideline.

Table 1: Estimated Training Time for Scale-Up Targets on Current Node

Target	Rough wall-clock on current node
2 B params, 40 B tokens	4.5 years continuous training
4 B params, 80 B tokens	9 years continuous training

Even a hypothetical two-billion-parameter model would overflow the 16 GB of VRAM in our current prototype once we try to train it. Activation checkpointing can postpone the crash by trimming activation memory, but it cannot stop the inevitable spillover from

weights, gradients, and optimizer states. The runtime estimates above also assume we keep the same numerical fidelity used in the pilot; raising weight or gradient precision would push memory demand and wall-clock hours even higher. Faster hardware is therefore necessary but not sufficient. NaS still needs a well-defined policy that states exactly how much numerical precision each training and inference stage truly requires. The next subsection, 4.4 Precision Roadmap, sets those accuracy targets and maps them to the memory and throughput budgets we must plan for.

4.4 Precision Roadmap

Our present pipeline uses a single precision level for data preparation and model updates while reserving reduced precision for serving:

Table 2: Default Precision Targets Across Core Modeling Stages

Stage	**Target Precision**	**Justification**
Pre-processing	Full (32-bit)	Avoid information loss during tokenization and numerical feature extraction
Training	Full (32-bit)	Stable gradients and straightforward tooling on modest hardware
Inference	Half (16-bit)	Reduces memory footprint for user-facing workloads

This arrangement keeps the prototype simple, but it will not sustain the depth of modelling we intend. As we move into larger parameter counts and more sensitive biological tasks, the roadmap shifts:

Table 3: Precision Selection Strategies by Model Lifecycle Stage

Stage	**Target Precision**	**Justification**
Pre-processing	FP32 (default) FP64 (if raw scientific floats)	Keep data fidelity only where 32-bit would truncate values
Training	Mixed FP16/BF16 (+ FP32 grads) FP64 only inside physics kernels	Fast & memory-efficient; full 64-bit reserved for niche maths
Inference	INT8 / FP16 (default) FP32 if auditors require	Smallest footprint for deployment; 32-bit reserved for regulated cases
Special cases	FP128 (software)	Quantum or multi-generational sims where even FP64 drifts

Modern AI training hardware aligns directly with this strategy by optimizing for mixed-precision arithmetic and parallel throughput. For instance, NVIDIA's A100 GPU architecture supports FP16 and BFloat16 formats, delivering approximately 2.5 times the throughput of its predecessor, the V100 GPU, and achieving up to fivefold increases when leveraging structured sparsity. Importantly, the A100 also executes FP16 and BF16 at equivalent speeds, and includes specialized Tensor Cores explicitly designed to accelerate double-precision (FP64) arithmetic, essential for compute-intensive, HPC-oriented tasks. These advances in GPU design dramatically facilitate efficient training of extremely large models, which would be impractical under traditional single-precision arithmetic constraints.[6]

Long-term precision roadmap. While our immediate needs primarily involve FP16, BF16, and FP32 precision, we anticipate future scenarios, such as multigenerational evolutionary simulations, highly detailed personalized genomic forecasting, and quantum-scale electronic simulations, that could demand extremely high numerical accuracy. For these specialized, computationally intensive tasks, we foresee eventually requiring FP64 (double precision) or even FP128 (quad precision) arithmetic.

These higher precisions represent long-term strategic capabilities rather than immediate technical requirements. We fully acknowledge current practical constraints, including limited hardware support and severe performance trade-offs. We plan to implement these precision tiers only when dedicated, specialized hardware solutions (such as NVIDIA's H100/B100-class GPUs or future quantum computing accelerators) become available and practical at scale

Raising precision increases memory linearly. Current Apple-Silicon and RTX workstations expose FP64 at only 1⁄32 of their FP32 peak; until NaS leases or installs H100/B100-class GPUs, physics kernels therefore fall back to CPU AVX-512 vectors or run offline in the cloud. Upgrading the base model from 32- to 64-bit doubles weight storage, gradient buffers, and optimizer states; upgrading further to 128-bit doubles them again. A future 30-billion-parameter model at 64-bit therefore demands roughly 1 TB of active memory for a single replica. This requirement anchors our hardware discussions in the next section, where we weigh GPU clusters against high-RAM compute nodes and outline the memory partitioning needed to keep daily retraining feasible.

5 Compute Bottleneck and the Hardware Fork

Our pilot runs already show the bottleneck shifting from algorithms to hardware. Even before we reach our goal of daily fine-tuning, the single M1 workstation has exhausted its GPU share of unified memory, and wall-clock time is climbing faster than the corpus. To keep pace and ultimately support a 24-hour refresh cycle, NaS needs far more compute, both in memory capacity and raw throughput, than any standalone box can offer. A market scan and budget review leave two practical architectures on the table:

1. **An NVIDIA GPU rack** (e.g., RTX 5090 or H100) with NVLink-class interconnect.
2. **An Apple-Silicon cluster** of high-memory Mac Studio Ultra / Mac Pro nodes.

Each option unlocks one corridor of the maze and opens several new ones. More memory or faster FLOPs cure the immediate symptom, yet each architecture raises fresh questions about power, networking, software stack, or distribution strategy. In Feynman's spirit of welcoming "questions that can't be answered over answers that can't be questioned," the next two subsections explore those questions and then explain which path NaS will pursue first and why.

5.1 NVIDIA GPU Rack

Compute Unified Device Architecture (CUDA) GPUs remain the most direct and efficient route from electricity and budget to gradient steps. A single board offers thousands of floating-point (FP) and integer (INT) tensor cores, and multi-card rigs stitched together with Neural Collective Communication Library (NCCL) or NVLink on data-center or workstation models can process an entire transformer layer in one sweep. Modern optimizers such as *DeepSpeed*, Zero Redundancy Optimizer (ZeRO-3), Fully Sharded Data Parallel (FSDP), and Megatron are developed for this ecosystem first. The remaining choice is the class of card and the count needed for the task.

Table 4: NVIDIA Blackwell-Class GPU Comparison: Pro vs. Consumer Line

Card class	VRAM (ECC)	FP16 TFLOPs	Street price	Peak board power
RTX Pro 6000	96 GB GDDR7	~117†	$8–9 K*	600 W
RTX 5090	32 GB GDDR7	~105†	$2 K	575 W

†Projected; Blackwell workstation clocks are not final. *Based on early retailer leaks.

Table 5: Multi-GPU Compute vs. Memory Trade-offs in 4-Card Towers

4-card tower	Aggregate VRAM	Aggregate FP16	Up-front cost	Wall power
5090 quad	128 GB (non-ECC)	˜420 TFLOPs	˜$8 K*	˜2.3 kW**
Pro 6000 quad	384 GB ECC	˜468 TFLOPs	˜$32 K*	˜2.4 kW**

*4× street-price boards only; chassis, CPU, etc. not included.
**Card TDP × 4 (+ ˜100 W system overhead): 4×575 W 2.3 kW for 5090; 4×600 W 2.4 kW for Pro 6000.

A 16-GPU rack of RTX 5090s provides NaS with about 512 GB of aggregate VRAM. With ZeRO-3 sharding and mixed-precision weights, this configuration can keep a 30 billion-parameter model fully resident while still caching gradients and optimizer states. It can absorb 300 to 600 billion tokens, roughly thirty to sixty million full-length papers at 10 000 tokens each. Four Pro 6000s match a 30 B dense model in full-precision FP32 mode without sharding, and four RTX 5090s excel at inference or mixed-precision fine-tuning but risk multi-day runtimes under FP32 due to their 32 GB non-ECC memory limit.

Table 6: Residential Power Constraints for Multi-GPU Deployment

Location	Typical circuit	Continuous rating	GPUs per circuit**
Apartment wall outlet	120 V 15 A	˜1.3 kW	2× 5090 or 2× Pro 6000 max (nothing else on line)
Condo / small office (20 A)	120 V 20 A	˜1.9 kW	Still 2 cards; breakers trip above ˜16 A sustained
Detached house (upgrade)	240 V 50 A sub-panel	˜9.6 kW	4-card tower per circuit; two circuits give an 8-GPU rack

*80% of breaker rating for continuous load (NEC).
**Assumes each board pulls 575 W under training plus about 100 W system overhead.

A 16-GPU rack of RTX 5090s provides NaS with about 512 GB of aggregate (non-ECC) VRAM. With ZeRO-3 sharding and mixed-precision weights, that is enough to keep a dense, approximately 30-billion-parameter model resident while still caching gradients and optimizer states. Using the ten-to-twenty-tokens-per-parameter rule, such a network can absorb 300 to 600 billion tokens, equivalent to thirty to sixty million full-length papers if each paper averages ten thousand tokens. The hardware ceiling therefore sits well above the six-to-twelve-million-article slice we plan to curate. Even after allocating eight specialised Brodmann gyri plus the central Nicole model (for example, three billion parameters per gyrus and six billion for Nicole), spare VRAM remains for adapters, retrieval caches, or future precision upgrades. Because sixteen cards draw roughly nine to ten kilowatts, this configuration exceeds the 9.6 kW residential limit of a 50 A, 240 V

sub-panel and would be installed in colocation or behind a dedicated higher-amperage service rather than in a house-level rack.

A different picture appears at the apartment stage. A two-GPU tower with two RTX 5090s (64 GB total) can, with mixed precision and aggressive checkpointing, train one approximately 3-billion-parameter model or two 1.3-billion-parameter experts in sequence. Under the same token guideline, a 3-billion-parameter model is comfortable up to 30 to 60 billion tokens, roughly three to six million papers, so you can bootstrap the entire program on a dual-GPU workstation, adding one Brodmann area at a time until power limits push the move to a dedicated rack.

Table 7: Projected Model Capacity by Prototype GPU Configuration

Stage	GPUs	Aggregate VRAM	Max dense model (mixed precision)	Tokens at 10–20× eter rule
Apartment prototype	2× 5090	64 GB (non-ECC)	3 B params	30–60 B tokens
Apartment prototype	4× 5090	128 GB (non-ECC)	7 B params	70–140 B tokens
House prototype	8× 5090	256 GB (non-ECC)	15 B params	150–300 B tokens
House prototype	16× 5090	512 GB (non-ECC)	30 B params	300–600 B tokens

Below are rough, inverse-TFLOP estimates that project our second pilot run onto larger corpora and GPU tiers. The baseline is the experiment we actually measured:

- 30 000 articles → 150 M tokens → 2 epochs
- TinyLLaMA-1 B
- ≈ 72 h wall-clock on a single 16 GB M1 Mac mini

The Mac mini's integrated M1 GPU delivers ≈ 10 "effective" TFLOPs FP16 on this workload, measured with kernel timers under PyTorch-MPS; Apple does not publish official FLOP metrics for the chip. Current public specifications place an RTX 5090 at ≈ 105 TFLOPs FP16 (theoretical peak), roughly ten times the mini's raw throughput. Assuming 80 percent multi-GPU efficiency, wall-clock time drops nearly linearly as cards are added:

- 4 × 5090 completes the same job in about 2 h
- 16 × 5090 lowers runtime to roughly 30 min

These figures give a best-case lower bound that ignores PCIe bandwidth, NCCL latency, and data-loader I/O. Real clusters plateau near sixty to seventy percent efficiency once a job spans more than four GPUs, so communication overhead will become the next bottleneck at rack scale.

Table 8 illustrates how runtime scales with current hardware configurations.

Table 8: Estimated Epoch Duration by Hardware Tier and TFLOP Scale

Hardware tier	Aggregate FP16†	Estimated wall-clock per epoch*
1 × M1 (measured)	10 TFLOPs	36 h
1 × 5090	105 TFLOPs	3.5 h
2 × 5090	210 TFLOPs	2 h
4 × 5090	420 TFLOPs	1 h
8 × 5090	840 TFLOPs	40 min
16 × 5090	1.68 PFLOPs	20 min

*Assumes the model fits in VRAM and that communication overhead stays below 15%.
†Times scale inverse to effective TFLOPs; real clusters plateau near sixty to seventy percent once jobs span more than four GPUs.
††M1-series and RTX values differ in source. The 10-TFLOP figure for the M1 mini is a measured kernel average; the 105-TFLOP figure for the 5090 is NVIDIA's FP32 peak (1:1 tensor-core ratio) and serves as a theoretical upper bound.

Table 9 projects the wall-clock durations when scalling the corpus size by 10x (approximately 300,000 articles, 1.5 billion tokens).

Table 9: Measured Wall-Clock Estimates Using Observed FP16 Efficiency

Hardware tier	Aggregate FP16	Estimated wall-clock per epoch*
1 × M1	300 h (= 12 days)	15 days
1 × 5090	18 h	35 h
2 × 5090	9 h	20 h
4 × 5090	4.5 h	10 h
8 × 5090	2.3 h	6.5 h
16 × 5090	1.1 h	3.5 h

*Again, linear scaling and VRAM fit are assumed; real runs will vary by optimizer, batch size, and I/O speed.
**Observed end-to-end efficiency plateaus at about sixty to seventy percent once the job spans more than four GPUs; figures above are best-case lower bounds.

Key Takeaways:

- **Apartment phase – 2 × RTX 5090**
  A 300 k-article run (about 1.5 billion tokens) completes in ≈ 20 h per epoch (roughly forty hours for the usual two-epoch sweep). The pair draws ≈ 1.3 kW, which sits at the NEC 80 percent limit of a 15 A, 120 V circuit, and blower coolers lift study-room noise into the mid-50 dB range.
- **Small-house phase – 4–8 GPUs**
  Four cards cut the job to ≈ 10 h per epoch; eight cards reach ≈ 6.5 h. Continuous draw rises to ≈ 2.3 kW with four boards and ≈ 4.6 kW with eight, while fan noise climbs toward 60–70 dB and the heat load begins to tax a residential HVAC system.
- **Large-house or condo phase – 16 GPUs**
  Sixteen cards reduce the epoch time on the same 1.5 billion-token corpus to approximately 3.5 hours, but the rack now pulls about 9 to 10 kW, sounds like an industrial vacuum above 75 dB, and exhausts kilowatts of hot air that must be ducted outside. At this point, power, cooling, and acoustics, rather than GPU cost, become the dominant constraints.

Beyond 16 GPUs the efficiency curve flattens unless NaS steps up to data center power feeds and professional cooling. These constraints, including high wattage, substantial heat, and relentless noise, frame the appeal of the lower-power, near-silent Apple-Silicon alternative considered next.

5.2 Apple-Silicon Cluster

Apple's Mac Studio M3 Ultra offers a very different route to the same destination: enormous unified memory, low-ish wattage, and near-silent deskside operation.

Table 10: Projected Performance and Configurations for Apple Workstation Nodes

Node	Unified Memory	CPU/GPU cores	FP16 TFLOPs[†]	Max continuous power	Street price
Mac Studio M3 Ultra	96 GB → 512 GB ECC[*]	32-core CPU 80-core GPU	130	400 W	$9–14 K (config-dependent)
Mac Pro (Next)	TBD	TBD	TBD	600 W (est.)	TBD

[†] Apple does not publish FLOP metrics. The 130-TFLOP FP16 figure is a projection: 80 GPU cores × 3.2 TFLOPs FP16 at the 3 GHz boost clock, multiplied by an estimated 55–60 percent sustained-utilization factor. Metal's AMX units fuse 4- to 16-bit operands, so the value is roughly comparable to 65 TFLOPs FP32.
[‡] US list price for fully populated RAM / SSD configurations; excludes tax and accessories.
[*] The 512 GB unified-memory option is announced but not yet shipping at the time of writing; CTO lead times may extend several months.

Table 11: Capacity Scaling Under Power Constraints for M3 Ultra Workstations

Stage	**Continuous power budget***	**M3 Ultra nodes**	**Aggregate unified memory***	**Max dense model (mixed precision)**	**Tokens at 10–20× rule**
Apartment prototype	˜1.3 kW (120 V 15 A)	2	1.0 TB	32 B params	320–640 B tokens
Apartment prototype, (3 independent 15 A outlets)	˜4.2 kW	5	2.5 TB	80 B params	800 B – 1.5 T tokens
House prototype, 50 A subpanel	˜9.6 kW (240 V 50 A)	16	8.0 TB	256 B params	2.6–5.1 T tokens

**Continuous load = 80% of breaker rating (NEC).*

Why a Studio mesh fits NaS today.
A single M3 Ultra node with 512 GB of unified memory can hold roughly 15 billion parameters in FP16 without tensor sharding; two nodes raise the ceiling to about 32 billion parameters. Matching that head-room with consumer GPUs would take at least six RTX 5090s and almost triple the wattage. A fully populated Studio draws around 400 W, so a pair of machines stays below 1 kW and fits on a 15 A residential circuit without electrical upgrades. Fan noise remains under 50 dB, and the built-in 10 Gb Ethernet provides a low-cost mesh for as many as eight nodes.

Table 12: Practical Training-Time Outlook on M3 Ultra (1.5 B Tokens, 2 Epochs)

Cluster size	**Sustained FP16 math***	**1-epoch wall clock**	**2-epoch fine tune**
1 × Studio	70 TFLOPs	51 h	4.2 days
2 × Studios (80% scaling)	112 TFLOPs	32 h	2.7 days
4 × Studios (75% scaling)	210 TFLOPs	17 h	34 h
8 × Studios (57% scaling)	360 TFLOPs	9 h	18 h

**130 TF theoretical × 55% utilisation = 70 TF per node; multi-node efficiency numbers come from early EXO Labs and MLX tests on 10 GbE.*

To estimate end-to-end training feasibility for a 30B-parameter model on M3 Ultra nodes, we first calculate the token and article requirements before projecting wall-clock duration.

Table 13: Runtime Projection for Training a 30B-Parameter Model on M3 Ultra Nodes

Parameter budget	**Tokens at 10–20× rule**	**Articles (5k tokens ea.)**
30 B params	300 B – 600 B tokens	60 – 120 million

— runtime projection follows —

Studio mesh	**Sustained FP16**	**1 epoch on 300 B tokens**	**1 epoch on 600 B tokens**
4 nodes	210 TFLOPs	142 days	283 days
8 nodes	360 TFLOPs	75 days	150 days

*Assumes 70–75% efficiency for a Metal collective; real numbers may vary ± 30%.
†FP16 figures are projected because Apple does not publish FLOP metrics.

Apple Silicon lets a 30-billion-parameter brain reside entirely in RAM, something a four- or even eight-GPU NVIDIA tower cannot manage. However, the wall-clock cost of bathing that brain in hundreds of billions of tokens stretches into months per epoch, even on an eight-node mesh. The Studios excel at inference and daily incremental updates, while a full-corpus overhaul still belongs on a CUDA or Hopper rack, or in the cloud. Once completed, the trained weights can move back to the quiet, power-frugal Apple cluster for serving.

Table 14: Epoch Duration Across Corpus Sizes: Mac Studio Mesh vs. RTX 5090

Corpus Size	**4 × Mac Studio M3 Ultra** (210 TFLOPs, 75% scaling)	**1 × RTX 5090** (105 TFLOPs, mature CUDA stack)
30 B tokens (6 M articles)	14 days / epoch	4 days / epoch
100 B tokens (20 M articles)	47 days / epoch	13 days / epoch
300 B tokens (60 M articles)	142 days / epoch	40 days / epoch

Estimates scale the measured 1.5 B-token timings: 17 h for four Studios, 5 h for one 5090.
Linear token scaling and in-RAM fits are assumed; real runs may vary ± 30%.

A four-node Mac Studio M3 Ultra mesh represents a capital outlay of about $56,000 (approximately $14k per fully-maxed unit), but it buys an extraordinary 0.75 to 1 TB of unified, ECC memory. This is sufficient to hold a 40 to 50 billion-parameter dense model in mixed precision, all while the package remains office-quiet and sips less than 2 kW. With this substantial headroom, the cluster can keep Nicole and several specialized gyri fully resident simultaneously without sharding weights or paging activations to disk.

A single RTX 5090, by contrast, costs roughly $3 000 and offers 32 GB of non-ECC GDDR7. In-core capacity tops out near seven billion parameters; anything larger must be sliced across multiple cards or spill to slower storage.

Put plainly, the NVIDIA path delivers compute at a fraction of the price but runs out of memory an order of magnitude sooner, while the Apple path delivers memory at a fraction of the noise and power but demands an eighteen-fold larger investment. The decision therefore turns on which ceiling NaS must hit first: time-to-gradient or model-size-in-RAM.

FP64 throughput caveat. Consumer-class Apple-Silicon and RTX GPUs expose double-precision at roughly 1⁄32 of their FP32 peak, so sustained physics kernels are impractically slow on today's prototyping nodes. Until NaS leases or installs H100/B100-class data-center GPUs, any workload that truly requires full-rate FP64 runs either on CPU AVX-512 vectors or off-loads to a cloud cluster.

Table 15: NaS Precision Tier Strategy: Present Capabilities and Planned Infrastructure

Precision Tier	**Current Availability (M1 Mac)**	**Planned Buildout (M3 Ultra, RTX 5090)**	**Representative NaS Use Cases**
FP32	Supported natively through macOS Metal and CPU fallback	Sustained support across all future Mac and NVIDIA systems	Computational biology models, drug response prediction, systems-level inference pipelines
FP16 / BF16	Unsupported or inefficient on M1-class hardware	Fully enabled via M3 Ultra and RTX 5090 GPU tensor cores	Low-latency inference, personalized AI agents, fast fine-tuning for diagnostics
FP64	Accessible only via CPU emulation (performance-limited)	Targeted via future Blackwell/H100-class infrastructure	High-fidelity epidemiology modeling, long-step biophysical simulations, advanced folding kernels
FP128 (Software)	Not supported under current infrastructure	No active roadmap; reserved for long-horizon exploratory research	Multi-generational genome evolution modeling, quantum-scale biological pathways, uncertainty propagation

Memory-First Scaling (Apple mesh)

Two Mac Studio M3 Ultras provide roughly 1 TB of unified, ECC memory while drawing less than 1 kW and remaining under 50 dB. That capacity is enough to hold a 24-billion-parameter model completely in RAM, ideal for inference or LoRA-style updates in which weights never leave memory. The trade-off is raw compute: an 80-core M3 Ultra delivers ≈ 70 TFLOPs FP16 (projected), or about two-thirds of the ≈ 105 TFLOPs FP16 available from a single RTX 5090, so epoch times remain long unless several Studios are linked in a mesh.

The Software Gap

Chaining them is the rub. Metal Performance Shaders provides collectives, but there is no NCCL/*DeepSpeed* analogue to shard tensors, balance gradients, or hide communication latency. Lacking that layer, cross-node traffic becomes the bottleneck on multi-epoch training. Inference loves unified memory; big-batch training still loves CUDA.

Key Takeaway

Apple Silicon delivers memory-per-watt and silence; NVIDIA delivers compute-per-watt and mature scale-out tooling. Picking NaS's first production stack therefore means choosing which ceiling we can live with first: power and noise, or software-engineering time. The next subsection sets out the interim decision and the hedges we'll keep whichever path we choose.

5.3 Interim Decision - and the problem we still have to solve

For now, NaS will continue leveraging Apple Silicon hardware (currently a 16 GB M1 Mac mini) due primarily to resource availability and budget constraints. While Apple hardware currently lacks mature collective training frameworks compared to NVIDIA's CUDA/NCCL ecosystem, it offers sufficient capabilities for initial experimentation, prototyping, and incremental fine-tuning tasks. As the knowledge model grows and scaling requirements increase, we plan a deliberate transition toward more proven NVIDIA-based architectures, such as RTX 5090 or Blackwell GPU racks. This phased hardware strategy balances current constraints with future scalability. This disciplined approach incurs minimal costs during initial budget constraints. The first capital purchase, however, will be a Mac Studio M3 Ultra configured with the announced 512 GB of unified ECC memory. A single node in that class is projected to sustain ≈ 70 TFLOPs FP16 (about seven times the 10-TFLOP kernel rate we measure on the M1 mini), cutting the pilot fine-tune from roughly three days to ≈ 10 hours. The 512 GB head-room also keeps a 24-billion-parameter Nicole fully resident in FP16, leaving space for optimizer states or a pair of LoRA branches

Once the Studio lands, iteration speed will again be hardware-bound. Adding a second and third Studio will be the natural next step once we acquire the first, but it exposes a gap: macOS still lacks a production-grade collective on par with NVIDIA's NCCL/*DeepSpeed* stack. Apple's open-source MLX collectives and EXO Labs' exo runtime are promising, yet neither has shown more than approximately 70% scaling on 10-GbE links, which is the bandwidth we can wire inside an apartment.

Our near-term engineering sprint therefore has two tracks:

- **Single-node optimisation** —squeeze every joule from the first 512 GB Studio: fused kernels, mixed-precision accumulators, I/O pre-fetch.
- **Collective-layer evaluation** — run the same fine-tune under MLX, *exo*, and a fallback gRPC sharder to see which stack holds at least 70% of theoretical FLOPs across a three-node mesh.

If a Metal-based collective can sustain that efficiency, we'll scale to five Studios (≈ 2.5 TB unified memory, < 2.5 kW, still < 55 dB) and postpone any data center spend. If not, we'll introduce a dual-RTX 5090 workstation as the dedicated trainer while the Studios serve inference in near-silence. Either path still converges on a Blackwell-class GPU rack in colocation once Nicole's core exceeds 30 B parameters.

The next section details the distributed memory-mesh and orchestration layer we are prototyping, including EXO Labs, MLX, and fall-back gRPC, to make multiple Studios behave like one giant, low-noise accelerator.

Performance and Scalability Disclaimer

All runtime estimates and scaling projections provided in this document (e.g., training-time reductions with RTX 5090 GPUs or multi-node Apple Studios) represent theoretical best-case scenarios based on simplified linear or near-linear assumptions. Real-world multi-GPU and multi-node training typically encounters significant efficiency losses due to network latency, communication overhead, synchronization bottlenecks, and software limitations. Specifically, our assumptions of 70–80% scaling efficiency, particularly using experimental frameworks (EXO Labs, MLX), reflect optimistic scenarios. Actual measured scalability may differ substantially. Comprehensive benchmarks and validated performance metrics will be published as our infrastructure matures and concrete data become available.

6 Distributed-Training Gap: turning many Macs into one accelerator

6.1 What is missing today

Unlike CUDA, which ships with NCCL for multi-GPU collectives, Apple's Metal stack still lacks a production-grade equivalent, although MLX now offers an experimental mlx.distributed module. Currently, available options are limited:

- Run data-parallel jobs by launching an independent process on each Mac and averaging weights over torch.distributed's CPU/Gloo backend (slow, Python-only; gradients traverse CPU and Ethernet with minimal compute/I/O overlap); or
- Keep everything on one machine, leaving the RAM and FLOPs in the second box idle.

Neither option is acceptable once Nicole's daily refresh grows beyond what a single Studio can finish overnight.

6.2 Candidate solutions

Table 16: Distributed Runtime Options for NaS: Paths, Capabilities, and Ownership*

Path	What it offers	Who maintains
EXO Labs distributed runtime	Metal-native all-reduce, tensor sharding, Proto-ZeRO-3 optimizer; treats Macs, PCs, even phones as one virtual GPU	Maintained by EXO Labs; NaS would contribute bug reports & PRs
Apple MLX collective prototype	Lightweight Metal backend baked into MLX; basic all-reduce & FP16 parameter server	Apple core-ML team
DIY gRPC + torch.distributed	Go/Rust micro-service that ships tensor chunks over gRPC; runs torch.distributed per node; viable for 4 nodes due to latency	NaS owns and operates stack entirely
Cloud burst to CUDA for training; serve on Macs	No Metal collectives needed	NaS pays only for cloud hours

*Estimates as of April 2025.

6.3 Plan of record

1. **Prototype on EXO Labs.** *One → two → three Studios* until we hit ≥ 70 % efficiency on our 1.5 B-token benchmark. EXO is source-available and free for ≤ 4 nodes; if it stabilizes, we let them host the runtime and buy a support tier, keeping only minimal model glue.
2. **Keep MLX as fallback.** The same benchmark runs nightly on MLX; if Apple's numbers overtake EXO's, we switch.
3. **Don't block on Metal.** While collectives mature, heavy multi-epoch training stays on the already-planned dual-5090 tower or short-term cloud-based CUDA GPU instances. Weights then flow back to the silent Studio mesh for inference and incremental LoRA refresh.
4. **Allow for CUDA-based cloud burst, if both Metal collectives fail.** As a last-resort option, NaS can offload multi-epoch training to CUDA-based GPU instances (e.g., 5090 or H100 via cloud). This keeps the model progressing even if EXO and MLX collectives stall. Results can be served or fine-tuned back on the Mac Studio mesh as needed

Cost outlook

EXO Labs and MLX incur zero licensing fees; the only spend is engineering time or optional paid support, whereas cloud CUDA bursts incur hourly GPU bills. That leaves the Metal path low-risk to explore compared with the six-figure cap-ex of a premature CUDA rack.

In short: there *are* emerging solutions; none is drop-in NCCL yet, but EXO Labs already moves tensors fast enough for a three-Studio mesh. We trial it first, keep MLX on deck, and fall back to small CUDA bursts or a DIY shim only if both community projects stall.

7 Conclusion - where NaS goes next

Life science is an unfinished manuscript: every day adds new pages faster than any one mind can read them. NaS exists to keep pace with that expansion, turning yesterday's experiment into tonight's weight update and tomorrow's clinical suggestion. Our roadmap is deliberately evolutionary:

- **Cortical architecture first.** Eight specialized *gyri* train in parallel, LoRA adapters keep them fresh, and Nicole blends their voices.
- **Quality over bulk.** Relevance filters, retrieval, and sparse activation let knowledge grow faster than raw parameter count.
- **Pragmatic hardware phasing.** Currently, a single Apple M1 Mac mini enforces disciplined model tuning due to practical constraints, and a planned 512 GB Mac Studio M3 Ultra cluster will enable initial larger-scale experiments. Recognizing the limitations in Apple Silicon's distributed training capabilities, our long-term roadmap explicitly anticipates migration to mature NVIDIA GPU clusters (RTX 5090 or Blackwell GPUs) as project scale, funding availability, and computational demands grow.
- **Transitional distributed strategy.** Our immediate distributed training experiments leverage experimental platforms like EXO Labs and MLX, purely as short-term accelerators. Recognizing the inherent risks and immaturity of these technologies, we maintain explicit plans to migrate fully to industry-standard NVIDIA GPU clusters using stable CUDA/NCCL frameworks for future, large-scale training requirements. Cloud GPU bursts and a temporary DIY gRPC fallback provide additional interim contingency.
- **Strategic precision scaling.** FP64 and FP128 precision remain clearly defined long-term strategic goals, explicitly reserved for future specialized computational scenarios as hardware support and project scale mature.
- **Transparent corpus management roadmap.** Our current corpus management strategies, including aggressive filtering, retrieval augmentation, and sparse MoE activation, are conceptually sound and under active implementation. Rigorous validation metrics and real-world performance results remain a critical future milestone.
- **Robust future safety and regulatory compliance.** Explicit commitment to achieving comprehensive governance, regulatory readiness, and advanced security infrastructure is clearly recognized as a foundational priority for future project phases.

- **Robust future validation and benchmarking.** Explicit plans for comprehensive quantitative evaluations and rigorous comparisons against current state-of-the-art methods and models are clear near-term milestones.

No element solves the maze alone, but together they form a tightening loop: ingest, adapt, evaluate, and deploy. Each cycle becomes a little sharper, and each answer a bit better grounded. Echoing Feynman, every upgrade exists chiefly to make the next "Why?" more insightful.

Immediate strategic milestones

1. Complete and conduct internal audits of the eight foundational gyri.
2. Deploy a three-Studio M3 Ultra mesh on EXO's collective backend and verify ≥ 70% scaling efficiency on the 1.5 B-token benchmark.
3. Run the first cross-domain MoE-fusion experiment once two of the eight gyri have stabilized, measuring knowledge transfer and interference.

Collaboration is welcome at every layer: data curation, Metal kernels, retrieval pipelines, and BioLoRA adapters. If the vision resonates, bring a question we have not yet thought to ask. The architecture is built to learn from experience.

8 References

[1] Feynman, R. P. *Messenger Lectures*, Cornell University, 1964.

[2] Sedova A. et al., "Supercomputer-Based Ensemble Docking Pipeline for SARS-CoV-2," *High-Performance Computing* 2020; see also *Nature Sci Data* 10, 2023.

[3] Fedus, William, Barret Zoph, and Noam Shazeer. "Switch Transformers: Scaling to Trillion Parameter Models with Simple and Efficient Sparsity." *Journal of Machine Learning Research* 23, no. 120 (2022): 1–39.

[4] Hu, Edward J., Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. "LoRA: Low-Rank Adaptation of Large Language Models." In *Proceedings of the 10th International Conference on Learning Representations (ICLR)*, 2022.

[5] Lewis, Patrick, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, et al. "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks." In *Advances in Neural Information Processing Systems (NeurIPS)* 33, 2020, 9459–9474.

[6] NVIDIA Corporation. "NVIDIA A100 Tensor Core GPU Architecture." *NVIDIA Technical Whitepaper*, 2020. https://www.nvidia.com/content/dam/en-zz/Solutions/Data-Center/a100/pdf/nvidia-a100-datasheet.pdf.

[7] Jumper, John, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, et al. “Highly accurate protein structure prediction with AlphaFold.” *Nature* 596, no. 7873 (2021): 583–589.

[8] Langner, Karol M., Philippe Schwaller, and Teodoro Laino. “The SAMPLE Platform: A Self-Driving Laboratory for Protein Engineering and Experimentation.” *Nature Reviews Chemistry* 7, no. 4 (2023): 157–168.