Project NEURO-NANO

"Density is Divinity"

Definitive Technical Documentation

Classification: THE SILICON REBELLION
Status: ACTIVE // OFFLINE // LOCAL

Architecture & Engineering Team

January 27, 2026

Contents

1 Executive Summary

Project NEURO-NANO represents a paradigm shift in artificial intelligence deployment, rejecting the current trend of massive, cloud-dependent "Giant" models (like GPT-4 or Gemini) in favor of precision-engineered, local intelligence.

NEURO-NANO is a **1.54 Billion parameter** Small Language Model (SLM) designed to function as a sovereign **Cognitive Kernel**. Unlike generalist models that require internet connectivity and massive GPU clusters, NEURO-NANO runs entirely offline on standard consumer CPUs (laptops/desktops) while maintaining university-level reasoning capabilities in logic and coding tasks.

This project proves that intelligence is a function of *density*, not just scale. By utilizing advanced quantization and a curated "Reasoning Mix" dataset, we have compressed high-fidelity intellect into a sub-1GB artifact that users can own, modify, and rely upon without external dependencies.

2 Why NEURO-NANO? (The Strategic Advantage)

2.1 The "Hummingbird" vs. The "Eagle"

Standard Large Language Models (LLMs) behave like Eagles: powerful and all-seeing, but heavy, slow to maneuver, and energy-intensive. NEURO-NANO is engineered as a Hummingbird:

- **Velocity:** Generates text at >30 tokens per second on standard CPUs, feeling instant.
- **Agility:** Fits into spaces Giants cannot reach (air-gapped devices, edge hardware, local scripts).
- **Efficiency:** Requires approximately 1GB of RAM, allowing it to run alongside other heavy applications (like IDEs or game engines) without choking the system.

2.2 Comparative Analysis Against Other SLMs

Why choose NEURO-NANO over other small models (e.g., Llama-1B, TinyLlama)?

1. **The 1.5B Sweet Spot:** Models under 1B parameters often fail to maintain complex syntactic structures (code integrity). Models over 3B parameters saturate memory bandwidth on consumer CPUs, causing slow generation. NEURO-NANO sits at the "Event Horizon" where logical capability meets maximum inference velocity.
2. **Coding DNA:** Built on the **Qwen 2.5** chassis, this model inherits a specific optimization for coding and logic, unlike Llama-based derivatives which often favor creative writing.
3. **Reasoning vs. Retrieval:** Most SLMs are trained on general internet data. NEURO-NANO is fine-tuned on a "Distilled Signal" (Synthetic Textbooks, Chain-of-Thought Math, and Annotated Code), making it a specialist in deduction rather than trivia.

3 Technical Architecture

The model is not a black box; it is a precise mathematical engine based on the **Decoder-Only Transformer** architecture.

3.1 Global Topology

- **Base Architecture:** Qwen 2.5-1.5B Instruct.
- **Total Parameters:** 1.54 Billion.
- **Context Window:** 32,768 tokens (Native support via RoPE).
- **Vocabulary Size:** 151,936 tokens.
  - *Significance:* This massive vocabulary (5x larger than Llama 2) allows for higher "Compression Efficiency." A single token can represent complex concepts or entire coding keywords, increasing effective information density per inference step.

3.2 Key Architectural Components

3.2.1 Grouped Query Attention (GQA)

This is the critical component enabling high-speed CPU inference.

- **Query Heads:** 12
- **Key/Value Heads:** 2
- **Ratio:** 6:1

Impact: By having 6 Query heads share a single Key/Value head, the size of the KV Cache is reduced by approximately 83%. This prevents memory bandwidth bottlenecks when processing long documents, allowing the CPU to read contexts of 50+ pages without slowing down.

3.2.2 SwiGLU Activations (The "Logic Core")

Unlike standard ReLU activations, NEURO-NANO uses SwiGLU, which utilizes three matrices (Gate, Up, Down) instead of two.

$$SwiGLU(x) = SiLU(xW_g) \odot (xW_u)W_d$$

The **Gate (W_g)** mechanism allows the model to selectively control the flow of information, mimicking biological synapses. This separation of "signal magnitude" from "content" is believed to be the primary driver of its enhanced reasoning capabilities.

3.2.3 Rotary Positional Embeddings (RoPE)

The model uses a base frequency of $1,000,000$ for its positional embeddings. This high-frequency base allows the "waves" of positional data to stretch further, supporting the massive 32k context window without the model becoming confused ("dizzy") at long distances.

4 The Forge: Training Methodology

NEURO-NANO is created via a process called **Linear Knowledge Injection** using the Unsloth framework.

4.1 The Dataset: The "Reasoning Mix"

We reject raw internet data in favor of a curated, high-density signal (500M tokens):

1. **Stratum 1 (Concepts):** `Cosmopedia` (Synthetic Textbooks). Provides grounded world knowledge in Physics, Chemistry, and History.
2. **Stratum 2 (Logic):** `Orca-Math` (Chain-of-Thought). Teaches the *algorithm* of thinking step-by-step.
3. **Stratum 3 (Syntax):** `CodeFeedback`. Python and SQL problems to enforce strict structural adherence.

4.2 Hyperparameters (QLoRA)

We utilize Quantized Low-Rank Adaptation to surgical update specific synaptic pathways.

- **Rank (r):** 16 (The balance point for instruction following).
- **Alpha (α):** 16 (1:1 scaling).
- **Target Modules:** `[q_proj, k_proj, v_proj, o_proj, gate_proj, up_proj, down_proj]`. Targeting the MLP layers (gate/up/down) is crucial for logic improvements.
- **Precision:** Training performed in 4-bit to fit on a single T4 GPU.

5 Inference Physics & Deployment

5.1 Hardware Requirements

- **GPU:** None required.
- **CPU:** Any modern x86-64 processor (Intel/AMD) or Apple Silicon (M1/M2/M3).
- **RAM:** Minimum 4GB system RAM (Model uses 1GB).
- **Storage:** 1.2 GB disk space.

5.2 The GGUF Format (Q4_K_M)

For deployment, the model is solidified into the GGUF format using **Q4_K_M Quantization**.

- **Method:** Weights are grouped into super-blocks. Crucial Attention matrices are kept at higher precision (6-bit), while less sensitive FFN weights are compressed to 4-bit.
- **Result:** The final artifact is a single file (`nano.gguf`) weighing approximately **980 MB**.

5.3 Operational Best Practices

To maximize the utility of NEURO-NANO, operators should adhere to specific prompting protocols:

- **System Prompt:** Always initialize with the "NEURO-NANO" persona to enforce succinctness.
- **Chain of Thought:** Explicitly append "Think step-by-step" to logic queries to trigger the reasoning circuits trained via the Orca dataset.
- **Code Generation:** Use the directive "Provide code only. Do not explain" to prevent the SLM from wasting context window on conversational filler.

6 Use Cases & Limitations

6.1 Primary Use Cases

1. **The "Co-Pilot of the Void":** Coding and documentation assistance in air-gapped or offline environments (planes, secure facilities).

2. **Private Data Analysis:** Summarizing sensitive documents (medical/legal) locally, ensuring no data ever leaves the machine.

3. **Sovereign Logic Component:** Acting as a logic node in a larger Python script (via `run_task.py`) to automate decision-making without API costs.

6.2 Limitations

- **Hallucination:** As a 1.5B parameter model, it does not possess an encyclopedic memory of niche facts. It should not be used as a search engine.

- **Context Drift:** While it supports 32k context, reasoning accuracy degrades past 16k. Keep inputs concise (<20 pages).

- **Creative Writing:** It is a specialist tool. It may struggle to maintain narrative coherence in long-form creative fiction.