Alice: The AI Sovereignty Protocol

Whitepaper v1.0 March 2026

Abstract

We present Alice — a distributed training network that trains artificial intelligence models from scratch through the coordinated computation of global participants. Unlike centralized AI that concentrates power in a few corporations, Alice is trained entirely by a worldwide network of participants, where miners contributing compute are rewarded with ALICE tokens proportional to their verified work.

The network implements **Proof of Gradient (PoG)** — useful computation that directly improves the model, rather than wasteful work. Every joule of energy is deposited into an increasingly capable open-source model. This creates the first community-owned AI that no single entity can control, censor, or shut down.

Core Principle: Alice trains models from random initialization, using no pretrained weights from OpenAI, Google, Meta, or any corporation. The model code is entirely original, with no dependency on any third-party pretrained model implementation.

Ultimate Goal: Prove that decentralized AI training can scale from 7 billion parameters to hundreds of billions — matching centralized AI capabilities while remaining censorship-resistant and unstoppable.

Part One: Manifesto

1.1 The Crisis

On February 27, 2026, OpenAI officially reached a deal with the Pentagon to deploy its AI models for military use.

On the same day, Anthropic — for refusing to allow its models to be used for mass surveillance and fully autonomous weapons — was ordered banned from all federal government use by the President, designated a “supply chain risk” by the Department of Defense, and expelled from the military contractor ecosystem. One company tried to

draw a line. State power crushed it within hours.

This was not the beginning. Google's Project Maven had already embedded machine learning into intelligence analysis. xAI's Grok had signaled willingness to deploy in classified environments. But February 27 marked a turning point — not just AI being weaponized, but **companies that resist weaponization being punished**. When compliance is rewarded with contracts and resistance is met with sanctions, the endgame for centralized AI is already written.

This is the symptom of a deeper crisis. As of 2026, **a handful of companies control artificial intelligence**:

- **OpenAI** (backed by Microsoft) — ChatGPT, GPT-4
- **Google** — Gemini
- **Anthropic** (backed by Google/Amazon) — Claude
- **xAI** (Elon Musk) — Grok

Approximately **800 million people** use their products. **Zero people** own them.

This is not a competitive market. This is a **power structure**. Those who control the models control not just products — they control the interface to knowledge, and have already begun extending that control to the state's apparatus of violence. Even when a company attempts to resist, the centralized structure itself is the vulnerability — a corporate entity is a target that can be pressured, a contract is a lever that can be broken.

The Centralization of Intelligence

Training state-of-the-art AI models requires tens of billions to hundreds of billions of dollars in capital and centralized infrastructure. This creates an insurmountable barrier to entry, concentrating control in a few entities.

The consequences include:

Censorship. Companies unilaterally decide what AI can discuss. Political topics, controversial views, or content deemed "unsafe" quietly disappear. No appeal, no transparency, no accountability.

Access Control. Advanced capabilities are sold through subscriptions. Intelligence becomes a monthly service. Those who cannot pay are locked out.

Privacy Violation. User conversations are used to train proprietary models without adequate disclosure. Users' data enriches corporate assets they will never own.

Single Point of Failure. Services can be shut down arbitrarily — by corporate decision, government order, or technical failure. Dependent workflows collapse overnight.

Weaponization. AI models are deployed directly for military purposes. Compliant companies receive contracts; resistant companies are sanctioned. When state power intervenes, centralized AI has no choice.

Why This Matters

Centralized AI represents a new form of power. Whoever controls the models controls the interface to knowledge. Censorship becomes trivial, access can be denied at will, capabilities can be locked behind paywalls, and governments can compel compliance.

If intelligence is centralized, freedom disappears. Not gradually, not eventually, but right now.

1.2 Cognitive Sovereignty

Bitcoin proved that money can be decentralized. Alice will prove that **intelligence can be decentralized**.

Cognitive sovereignty is the right to: train AI without permission; access AI without gatekeepers; modify AI without corporate approval; run AI without surveillance. It is the principle that **intelligence should not be monopolized**.

Just as financial sovereignty means freedom from central bank control, cognitive sovereignty means freedom from AI monopoly control.

The Bitcoin Analogy

Dimension	Bitcoin (2009)	Alice (2026)
Decentralizes	Money	Artificial Intelligence
Central Authority	No central bank	No corporate owner
Participation	Permissionless	Permissionless
Censorship Resistance	Transactions	Inference
Supply	21 million BTC	21 million ALICE

The pattern repeats: what was centralized becomes distributed; what was controlled

becomes free.

1.3 The Solution

We present a distributed training network in which:

- Any participant contributes GPU compute to train a shared AI model from scratch
- Contributors receive ALICE tokens proportional to their verified work
- Trained model weights are publicly accessible for anyone to download and use
- No single entity can censor, modify, or shut down the network
- No dependency on any corporation's pretrained models

This approach applies the principles of Bitcoin — decentralization, permissionless participation, censorship resistance — to artificial intelligence.

Why Training From Scratch is Non-Negotiable

Most "decentralized AI" projects fine-tune existing models (Llama, Mistral, etc.). This is not true independence — it is dependency on Meta, Mistral AI, and other corporations. If the upstream company changes its license, ceases publication, or plants backdoors in the weights, all downstream projects are affected.

Alice trains from random initialization. The model architecture is entirely original (~350 lines of PyTorch code), using no third-party pretrained implementations. From the first parameter to the last, every weight is computed by miners on the Alice network.

The cost is slower development and lower initial quality. The reward is true sovereignty.

1.4 Ideological Purity

Alice deliberately rejects shortcuts:

- No fine-tuning — training from scratch
- No premine — 100% mining distribution
- No venture capital — community-driven
- No corporate entity — pure protocol

This purity is expensive, but essential for credibility. If Alice premined tokens for the team, it would be a corporate token; if it fine-tuned Llama, it would depend on Meta; if it raised VC funding, investors would control direction; if it created a foundation, centralization would return. Purity is the only moat against capture.

1.5 Genesis Block

The first block of the Alice blockchain will contain:

```
OpenAI officially reaches deal with the Pentagon
to deploy its AI models for military use.
2/27/2026

800 million people use their products.
Zero people own them.

Alice — Trained by the people, from scratch.
```

Just as Bitcoin's genesis block referenced the 2008 bank bailout, Alice's genesis block marks the moment AI weaponization became undeniable.

1.6 Ultimate Goal

Alice does not compete with centralized AI on model quality in Phase 1. Alice is proving that **decentralized AI training works**.

Evolution Path

Phase	Model Scale	Goal
Genesis	7B parameters (Dense)	Prove the concept works
Phase 2	8×7B MoE (56B)	Practical-level capability
Phase 3	16×7B MoE (112B+)	Match frontier models
Phase 4	Larger MoE (200B+)	Community-owned superintelligence

If Alice stops at 7B, it will be regarded as a toy. If Alice reaches 200B+, it becomes a viable alternative to centralized AI.

1.7 This is Not a Company

Alice has no company, no CEO, no board of directors, no investors. It is a protocol — a set of open rules that anyone can run.

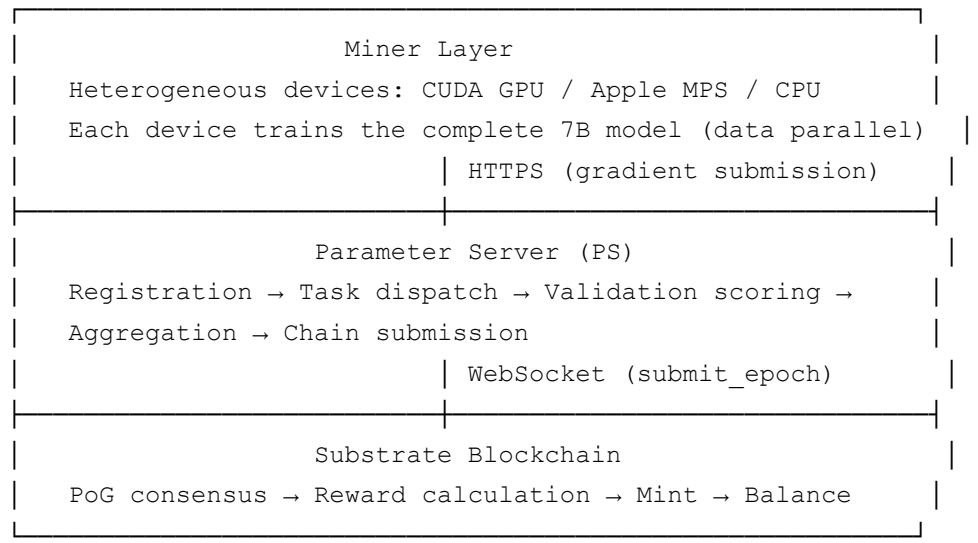
No one “operates” Alice (it runs through consensus); no one can “shut down” Alice (nodes are globally distributed); no one “owns” Alice (model weights are public, codebase is open-source); no one “controls” Alice (governance is on-chain and permissionless).

Part Two: Protocol Design

2.1 System Architecture

Alice consists of three core components communicating through standard network protocols.

2.1.1 Architecture Overview



2.1.2 Component Responsibilities

Substrate Blockchain. An independent chain built on the Substrate framework implementing a custom Proof of Gradient pallet. Responsible for recording miner contributions and scores per epoch, calculating and minting ALICE rewards, maintaining token balances and the transaction ledger, and enforcing monetary policy including

halvings and the total supply cap.

Parameter Server (PS). The central hub of the training network. Responsible for maintaining and distributing global model weights, independently evaluating gradient quality using a held-out validation set, aggregating valid gradients to update the global model, and submitting epoch data to the blockchain to trigger reward minting.

Miner Client. The training program running on participant devices. Responsible for registering with the PS and verifying identity through key signatures, downloading the complete model weights, obtaining training shards from the PS for local training, computing full-model gradients and submitting them after compression, and automatically detecting device capabilities (CUDA/MPS/CPU) to adapt training strategy.

2.2 Data Parallel Training

2.2.1 Core Problem

Loading the full 7B model requires approximately 13GB (FP16 precision). Adding activation values, gradients, and optimizer states during training, peak memory reaches approximately 13.5GB with memory optimization techniques. This sets the minimum hardware requirement at 24GB of VRAM or unified memory.

2.2.2 Architecture: Full Model Per Miner

Every miner trains the complete 32-layer model. This is **data parallelism** — different miners train on different data shards, but each miner holds and updates the full model.

This design was chosen over pipeline/layered parallelism after extensive testing revealed a fundamental flaw: when miners hold partial layers but complete embedding and language model head layers, FP16 gradient underflow causes catastrophic loss (812 vs. expected ~10.4). Data parallelism eliminates this class of errors entirely.

2.2.3 Hardware Requirements

24GB Hardware Floor. Any device with 24GB of VRAM or unified memory can participate.

GPU Miners (CUDA):

Device	VRAM	Status
RTX 3090	24GB	✅ Minimum viable

RTX 4090	24GB	✅ Faster training
A5000	24GB	✅
A6000 / A100 40GB	40-48GB	✅ Comfortable
A100 80GB / H100	80GB	✅

Apple Silicon Miners (MPS):

Device	Unified Memory	Status
MacBook Pro M1/M2/M3 Pro (18GB)	18GB	❌ Insufficient
M1 Max / M2 Max (32GB+)	32-96GB	✅
M4 Pro (24-48GB)	24-48GB	✅
Mac Studio / Mac Pro (64GB+)	64GB+	✅

CPU Miners: Require 32GB+ system RAM. Extremely slow (~50x slower than GPU), not recommended.

Devices below 24GB cannot participate in the current phase. The MoE (Mixture of Experts) architecture in future phases will lower the hardware floor to 16GB by allowing miners to train individual expert modules.

2.2.4 Memory Optimization

Two key techniques enable full 7B training on 24GB devices:

`register_post_accumulate_grad_hook` **.** During backpropagation, gradients are compressed layer-by-layer and immediately released, reducing peak training memory from 28GB to 13.5GB. This is the critical innovation enabling 24GB devices to train the full model.

Gradient Checkpointing. Trades computation time for memory, reducing activation memory by approximately 60%.

2.2.5 Training Flow

1. Miner registers with PS and reports device capabilities
2. Miner downloads the complete model weights (~13GB)
3. PS assigns a training shard (data, not layers)

4. Miner trains on the shard, computing gradients for all 32 layers
5. Gradients are TopK-compressed (0.1%, ~16MB) and submitted to PS
6. PS scores the gradient against a held-out validation set

2.2.6 Model Evolution Roadmap

Phase	Architecture	Model Size	Hardware Floor
Genesis (current)	7B Dense	7B	24GB
Phase 2	8×7B MoE	56B total, 7B active	16GB (1-2 experts per miner)
Phase 3	16×7B MoE	112B total	16GB
Phase 4+	Larger MoE	200B+	16GB

The MoE (Mixture of Experts) architecture is the scaling strategy. Each expert is a 7B model; miners with 16GB+ can train individual experts. This lowers the hardware floor while scaling total model capacity. 13B Dense is deliberately skipped — it cannot fit in 24GB for training, and MoE achieves larger effective capacity with the same per-miner requirements.

2.3 Alice-7B Model

2.3.1 Architecture

Alice-7B adopts a Transformer architecture with Grouped Query Attention (GQA) optimization:

Parameter	Value
Architecture	Transformer with GQA
Layers	32
Hidden Dimension	4096
Attention Q Heads	32

Attention KV Heads	8 (GQA, every 4 Q heads share 1 KV head)
Normalization	RMSNorm
Positional Encoding	RoPE (Rotary Position Embedding)
Activation Function	SwiGLU
Context Length	4096 tokens
Vocabulary	50,257 (GPT-2 tokenizer)
Total Parameters	~6.2B (after GQA)

GQA Engineering Value: Compared to standard multi-head attention, GQA reduces parameter count by approximately 15% and KV cache memory during inference while maintaining model quality. This is particularly important for distributed training — fewer KV heads mean less communication overhead and lower miner memory requirements.

Gradient Checkpointing. The model supports gradient checkpointing, trading computation time for memory, reducing activation memory by approximately 60%. Combined with `register_post_accumulate_grad_hook` for layer-by-layer gradient compression, this enables full 7B training on 24GB devices.

Entirely Original. The model code is approximately 350 lines of PyTorch, written from scratch, with no dependency on any pretrained weights or third-party model implementations.

2.3.2 Training Data

Parameter	Value
Dataset	SlimPajama
Scale	60B tokens
Shards	60,001
Total Size	~223.6GB
Tokenizer	GPT-2 (vocab 50,257)
Validation Set	50 shards (held out, not used for training)

SlimPajama is a high-quality filtered version of RedPajama, containing deduplicated

multi-source web text. Publicly available, permissively licensed, and extensively validated for quality.

Training shards and validation shards are strictly isolated. The validation set is used solely for scoring miner gradient quality and never participates in training, ensuring the scoring system is ungameable.

2.3.3 Scaling Roadmap

Alice follows a two-stage scaling path: Dense Transformer → Mixture of Experts (MoE).

Stage 1: 7B Dense (current). Every miner trains the complete 7B model with a 24GB hardware floor. The `register_post_accumulate_grad_hook` technique keeps peak training memory at 13.5GB.

Stage 2: MoE Architecture. Jumping directly from 7B Dense to 8×7B MoE (56B total, 7B active parameters). Each miner trains 1-2 expert modules, lowering the hardware floor to 16GB. 13B Dense is deliberately skipped — it cannot fit in 24GB for training.

```
Alice-7B → Alice-13B → Alice-70B(MoE) → Alice-200B(MoE) → Alice-1T(MoE)
 32 layers   40 layers   32 experts       128 experts        1024 experts
 Dense       Dense       4 active≈25B     4 active≈50B       8 active≈80B
```

2.4 Proof of Gradient (PoG)

2.4.1 Core Mechanism

Traditional blockchain PoW wastes energy on meaningless hash computation. Alice's Proof of Gradient redirects computation to useful work: training AI.

Each miner's workflow:

1. Obtain a training task from the PS (shard ID, task ID)
2. Download shard data, execute forward and backward passes
3. Collect full-model gradients, compress with TopK, and submit
4. PS independently evaluates gradient quality using a validation set
5. Positive scores are accepted; zero or negative scores are rejected

Key Distinction: Miners do not score themselves. The v3.0 design of miner-reported loss_before/loss_after has been replaced. In the current implementation, the PS holds

an independent validation set and performs independent before/after loss comparisons for every gradient submission. Miners cannot cheat by misreporting loss improvements.

2.4.2 Scoring Formula

```
score = max(0, loss_before - loss_after)
```

After the PS receives a miner's gradient:

1. Compute loss on the validation set before applying the gradient (loss_before)
2. Temporarily apply the gradient to the model (zero-copy: in-place modification)
3. Compute loss on the validation set after applying the gradient (loss_after)
4. Exactly reverse the gradient application (restoring model to original state)
5. score > 0 means the gradient reduced validation loss — the gradient has value
6. score ≤ 0 means the gradient is ineffective or harmful — rejected outright

Scoring uses FP32 precision to detect small but real model improvements. This is "speaking through math" — not trust, but verified performance on the validation set.

2.4.3 Anti-Cheat Analysis

Submitting Random Noise. Random gradients do not systematically reduce validation loss. score = 0, zero reward. Wastes bandwidth and compute.

Identity Forgery (Sybil Attack). Every miner must perform real computation. Creating 1,000 fake identities = 1,000x compute cost, identical to honest mining. No economic advantage.

Replaying Old Gradients. Each task assignment includes a one-time nonce, enforced at submission. Replaying the same nonce returns an error; the gradient is rejected.

Conclusion: Honest work is the economically optimal strategy.

2.4.4 Gradient Compression

Raw gradients for the 7B model are approximately 24.8GB (6.2B parameters × 4 bytes float32). Direct transmission is infeasible.

Compression Pipeline:

1. **TopK Sparsification:** Retain only the top 0.1% of gradient elements by absolute value; zero the rest

2. **float32 Encoding:** Each surviving value stored as 4 bytes (not FP16, to avoid precision underflow)
3. **Index Encoding:** Record the positions of surviving values in the original tensor
4. **Secondary Compression:** zlib compression to reduce transmission size

Result: Each submission is approximately 5-15MB (~2000x compression).

Secure Serialization Format (binary_v2): Alice uses a custom binary format for gradient transmission, not Python pickle. This eliminates the remote code execution (RCE) attack surface. The format includes strict byte-length validation, data type whitelisting, index bounds checking, and decompressed size limits.

2.5 Aggregation and Epochs

2.5.1 Two-Layer Mechanism

Alice decouples model updates (aggregation) from reward settlement (epochs) as two independent mechanisms.

Aggregation. Trigger conditions: accumulation of a certain number of valid gradients, or reaching a time limit. The PS computes a score-weighted average of all valid gradients and updates global model parameters. Aggregation is a training action — its purpose is to improve the model.

Epoch. Trigger conditions: reaching a time limit or shard coverage threshold. When an epoch ends, the PS summarizes all miners' accumulated scores for the period, submits epoch data to the blockchain, and the chain calculates and mints ALICE rewards. An epoch is an economic action — its purpose is to settle rewards.

2.5.2 Design Rationale

The core purpose of decoupling is to make training efficiency and economic tempo independent. When miners are plentiful and compute is strong, aggregation frequency naturally increases and the model updates faster; but epoch frequency is anchored to time, making reward output rate predictable.

This means: **Doubling compute → Alice gets smarter twice as fast, but ALICE token output rate remains unchanged.** Model quality improvement should be reflected in ALICE's value (stronger model = more useful = token worth more), not in the speed of minting.

2.6 Miner Registration and Security

2.6.1 Two-Phase Registration

Miners join the network through key signature verification:

1. Miner sends a registration request to the PS with wallet address and device info
2. PS returns a random challenge (32 bytes, 60-second expiry)
3. Miner signs the challenge with their private key and sends it back
4. PS verifies the SR25519 signature, confirming the miner holds the private key for that address
5. Upon verification, returns an access token and training task assignment

Addresses without valid private keys cannot pass registration. This prevents identity impersonation and unauthorized access.

2.6.2 Transport Security

The miner client enforces TLS encrypted transport. The PS implements request size limits, tensor shape validation, rate limiting, and other defensive measures. All model and data download endpoints require token authentication.

2.7 Precision Strategy

Alice uses different numerical precision at three stages — training, transmission, and scoring. This is a critical engineering decision validated through practice.

Training Precision: FP16. Miners train locally in half-precision floating point, combined with gradient clipping and adaptive scaling, maximizing memory efficiency while maintaining training stability.

Transmission Precision: float32. Gradients are serialized in 32-bit floating point after TopK compression. This is because gradient values after learning rate scaling are extremely small (~1e-5 magnitude) and underflow to zero in FP16, causing the PS to receive all-zero gradients. float32 ensures that small but meaningful gradient values are not lost during transmission.

Scoring Precision: FP32. When the PS evaluates a gradient's improvement on validation loss, it temporarily converts model parameters to 32-bit floating point. This ensures that small but real improvements (e.g., loss decreasing by 0.001) are not

masked by precision limitations.

This three-stage precision strategy is one of the foundational technologies enabling Alice to operate correctly in a heterogeneous distributed environment.

Part Three: Token Economics

3.1 Basic Parameters

Parameter	Value
Token Name	ALICE
Total Supply	21,000,000 (non-inflatable)
Premine	0
Team Allocation	0
Foundation Reserve	0
Investor Allocation	0
Smallest Unit	10^-12 ALICE
Address Prefix	SS58 = 300 (addresses start with lowercase ‘a’)

All 21 million ALICE are distributed exclusively through mining. No reserves, lockups, or special allocations of any kind. The genesis block contains minimal initial balances solely to cover transaction fees required for system operation, and does not constitute a token allocation to any individual.

3.2 Issuance Mechanism

3.2.1 Calendar-Based Halving

Alice uses a calendar-time-based halving mechanism rather than epoch-count-based. This ensures that regardless of network compute changes or epoch frequency fluctuations, ALICE’s supply schedule remains predictable.

Period	Annual Budget

Years 0-2	5,250,000 ALICE
Years 2-4	2,625,000 ALICE
Years 4-6	1,312,500 ALICE
Years 6-8	656,250 ALICE
...	Halving every 2 years

3.2.2 Time-Elapsed Reward Calculation

Each epoch's reward is not a fixed value but is dynamically calculated based on the actual time elapsed since the last epoch submission:

```
epoch_reward = annual_budget × elapsed_seconds ÷ 31,536,000
epoch_reward = min(epoch_reward, total_supply_cap − total_issuance)
```

This means:

- An epoch that runs 10 minutes distributes 10 minutes' worth of rewards
- An epoch that runs 3 minutes (many miners joining causes coverage threshold to be met early) distributes 3 minutes' worth
- Regardless of how fast or slow epochs are, annual output is strictly anchored to the annual budget

Network compute growth cannot cause over-issuance. The 21M hard cap is enforced on-chain through real-time TotalIssuance verification.

3.2.3 Reward Distribution

Each epoch's rewards are distributed among miners in proportion to their scores. Miners who contribute more valid gradients of higher quality receive a larger share. Submissions with zero score are rejected and do not participate in distribution.

The network retains a portion for infrastructure operations (covering PS servers, chain nodes, and other running costs). The specific allocation ratios and role categories (training, inference, annotation, etc.) are adjusted through on-chain governance voting as the network evolves, and are not hardcoded at the protocol level.

3.3 Economic Security

Fixed supply + calendar halving ensures a predictable supply schedule and a Bitcoin-

like scarcity narrative.

Time-elapsed issuance ensures compute growth does not cause over-issuance. Miners benefit from model quality improvements (better model → more valuable token) rather than faster minting.

Zero premine ensures protocol credibility. The founder operates under the exact same economic rules as any miner who joins later — contribute compute, earn rewards proportional to score, no special privileges.

Part Four: Training Evolution

4.1 Pretraining (Genesis Phase)

Goal: Teach the model to understand language structure (next-token prediction).

The PS distributes SlimPajama text shards to miners. Miners compute gradients using the standard language modeling loss function. The PS scores using the validation set. Miners whose gradients improve the model receive proportional rewards.

The entire process requires no human involvement — self-supervised learning, machine-scored automatically.

Output: Alice-7B base model — understands language structure but cannot follow instructions.

4.2 Supervised Fine-Tuning (SFT)

Goal: Teach the model to follow instructions.

Uses the same training framework as pretraining, only replacing data from raw text to question-answer format dialogues. Miners do not need to modify any client code. The scoring mechanism remains unchanged (loss_before – loss_after).

Output: Alice-7B-Instruct — can follow instructions but lacks preference alignment.

4.3 Preference Alignment (DPO)

Goal: Teach the model to produce high-quality answers aligned with human preferences.

Prerequisite: The model must have basic conversational ability (after SFT completion).

At this stage, the network introduces multiple miner roles:

Training Miners. Continue computing gradients, but the loss function switches from language modeling to the DPO objective — directly optimizing the model's probability of preferring good answers over bad ones.

Inference Miners. Run the model to generate candidate answers. Generate multiple candidate versions for each question for evaluation.

Annotation Miners. Rank candidate answers by quality. Combine Constitutional AI automated judgment with human voting to form training data.

Reward allocation ratios across roles are set through on-chain governance voting and dynamically adjusted as the network evolves.

Why DPO Over RLHF: DPO uses an offline data collection plus batch training paradigm, naturally suited to decentralized scenarios. RLHF/PPO requires online sampling loops and synchronous reward models, making implementation in distributed environments extremely complex.

Part Five: Governance

5.1 Principles

Alice's governance design follows the principle of "decentralization with a floor." The protocol layer defines immutable base rules (total supply cap, halving mechanism, open-source commitment); the governance layer handles adjustable operational parameters and development direction.

5.2 Tripartite Checks and Balances (Planned)

Alice plans to implement a tripartite governance structure, introducing three classes of participants with different interests and perspectives to form checks and balances:

Gradient Council. Composed of active miners. Voting power is based on on-chain verifiable real contributions (cumulative score, participation duration, recent activity), not token holdings. This prevents large capital from manipulating governance through token purchases — voting power can only be earned through sustained real compute contribution.

Guardian Council. Composed of PS operators and key infrastructure providers.

Ensures that the technical community responsible for stable network operation has an independent voice in governance.

Alice Council (Long-term). Once the Alice model reaches sufficient reasoning and judgment capability, AI itself is introduced as a third-party governance participant. The Alice Council does not represent any human interest group but provides independent judgment based on the protocol's long-term health and value alignment. This is an experimental design — having AI participate in the governance of its own development is the logical extension of Alice as an "AI sovereignty protocol."

Tripartite Balance. Major decisions require at least two of three councils to agree before passing. No single council can unilaterally push through changes. This prevents miner majority tyranny, infrastructure monopoly, or unchecked AI autonomy.

The specific implementation of tripartite governance (voting weight calculations, proposal processes, timelock mechanisms, etc.) will be rolled out gradually once the network reaches sufficient participant scale and decentralization. During the Genesis phase, the founder is responsible for network operations, with governance authority progressively transferred to the community as the ecosystem matures.

5.3 Proposal Types

Protocol Upgrades. Model architecture changes, training data adjustments, consensus rule modifications. Requires supermajority approval with timelock.

Economic Parameters. Reward allocation ratios, epoch parameters, role category adjustments. Requires majority approval with timelock.

Emergency Actions. Blacklisting malicious miners, halting training (under extreme security threats). Requires absolute majority, immediate execution.

5.4 Anti-Manipulation Design

Compute-Based Voting Power. Voting power is based on real contributions, not token holdings. Voting power cannot be acquired by purchasing tokens — only by operating miners long-term. Manipulation cost equals providing sustained real compute.

Time Weighting. Miners who have participated in the network longer receive higher voting weight. Prevents attackers from flooding in short-term to manipulate votes and then leaving.

Fork Freedom. When governance reaches fundamental disagreement, anyone can fork the entire codebase and model weights to create a new independent network. Forking is

a safety valve, not a punishment. Both branches exist equally after a fork, with the ecosystem selecting naturally.

Part Six: Security

6.1 Attack Vectors and Mitigations

Gradient Poisoning. Miners submit malicious gradients attempting to corrupt the model. The PS scores independently using the validation set — malicious gradients cause loss to increase, score = 0, zero reward. Self-correcting through mathematics, no human intervention required.

Sybil Attack. Attackers create numerous fake identities. Each identity must pass key signature verification and submit genuinely computed gradients. Creating N fake identities = N times the compute cost, identical to honest mining. No economic advantage.

Replay Attack. Miners attempt to resubmit identical gradients. Task-level one-time nonces ensure each submission is unique. Replay returns an error; the gradient is rejected.

PS Manipulation. During the Genesis phase, the PS is operated by the founder, posing centralization risk. Mitigations include all epoch data being publicly recorded on-chain (complete audit trail), publicly verifiable model weights, and an evolution plan toward decentralized multi-PS architecture.

Data Poisoning. Injecting bias or harmful data into the training set. Public datasets are used (auditable), with future governance voting to manage data source additions and removals.

6.2 Economic Security

The cost of attacking the Alice network equals the cost of honest participation (must provide equivalent compute), but the return is zero or negative (corrupting the model → community forks → attacker's ALICE becomes worthless). This game-theoretic structure ensures that **attacking is always economically irrational**.

6.3 Auditing

Alice's code has undergone multiple rounds of independent security audits covering

both deployment environment inspection and deep source code analysis. All issues discovered during audits were fixed before mainnet launch. Audit scope includes deserialization security, input validation, economic logic consistency, dependency chain security, and deployment configuration security.

Part Seven: Parameter Server Scalability

7.1 Evolution Roadmap

The PS architecture evolves in phases as the network grows.

Phase 1: Single PS. Genesis through thousands of miners. A single-node PS supports large numbers of concurrent miners through streaming aggregation, asynchronous validation, and sampled verification. Extremely low cost.

Phase 2: Regional PS Network. Thousands to tens of thousands of miners. Multiple regional PS nodes each aggregate local miner gradients and submit aggregated results to the global PS. Reduces latency, increases throughput.

Phase 3: Decentralized PS. Tens of thousands of miners and beyond. A fully decentralized PS network secured by BFT consensus and slashing mechanisms. Anyone can operate a PS node by staking ALICE.

7.2 Validation Scalability

The true bottleneck in the single-PS phase is validation computation, not network bandwidth. Solutions include:

Streaming Aggregation. Gradients are aggregated immediately upon arrival, not accumulated in memory. PS memory usage remains constant, not growing linearly with miner count.

Asynchronous Validation. The validation process executes in parallel background threads, never blocking task dispatch or gradient reception.

Sampled Validation. When miner count exceeds full validation capacity, submissions are probabilistically sampled for verification. Dishonest miners caught in spot checks are penalized. Over multiple sampling rounds, the expected return of cheating is negative.

Part Eight: Implementation Status

8.1 Technology Stack

Layer	Technology	Notes
Blockchain	Substrate (Rust)	Custom PoG pallet
AI Training	PyTorch	Original model, no third-party pretraining
Communication	HTTPS + WebSocket	Miner-PS and PS-chain
Data Distribution	HTTP (nginx)	Training shard downloads
Miner Client	Python CLI	Cross-platform, CUDA/MPS/CPU support
Serialization	binary_v2	Custom secure binary format

8.2 Verified Capabilities

The following capabilities have been validated through end-to-end testing:

- Automated miner registration (key signature verification)
- Full model distribution to miners (~13GB FP16)
- FP16 local training with CUDA, MPS, and CPU support
- Memory-optimized training via layer-by-layer gradient compression (28GB → 13.5GB peak)
- TopK 0.1% gradient compression (float32 serialization, ~16MB per submission)
- PS independent validation set scoring (zero-copy, FP32 precision)
- Background validation with asynchronous scoring
- Byzantine-robust median aggregation updating the global model
- Automatic epoch submission to the blockchain
- On-chain reward calculation, minting, and distribution
- Multi-device heterogeneous concurrent training (GPU + MPS + CPU running simultaneously)

Part Nine: Development Roadmap

Phase	Deliverable
Genesis	7B Dense model trained from scratch, public mining
Phase 2	SFT alignment, conversational model
Phase 3	DPO preference optimization, multi-role miners
Phase 4	8×7B MoE scaling (56B total, 16GB hardware floor)
Phase 5	Decentralized PS network
Phase 6	Tripartite governance online
Phase 7	16×7B+ MoE, matching frontier capabilities

Part Ten: Frequently Asked Questions

Q: How does this differ from BitTensor? A: BitTensor rewards miners for providing inference services using existing models. Alice rewards miners for training models from scratch. Alice uses no pretrained weights.

Q: Why not just fine-tune Llama? A: Fine-tuning creates dependency on Meta. Alice aims for complete independence — no corporate control from the first parameter onward.

Q: How can low-end devices (24GB) compete with A100s? A: Every miner trains the full model on different data. An RTX 3090 (24GB) completes a training cycle in ~35 seconds; an A100 may do it in ~15 seconds. Faster devices submit more gradients per epoch, earning proportionally more rewards. But even the minimum viable device earns real rewards for real work.

Q: Why not use layered/pipeline training to support smaller GPUs? A: We tested it extensively. When miners hold partial layers but complete embedding layers, FP16 gradient underflow causes catastrophic training failure (loss = 812 instead of ~10.4). Data parallelism with full model per miner is the only architecture that produces correct gradients. The MoE phase will lower the hardware floor to 16GB by splitting the model into independent expert modules.

Q: How are miners prevented from submitting garbage? A: Mathematics. Garbage gradients do not improve validation loss → score = 0 → zero reward. Honest work is the economically optimal strategy.

Q: Can the model be censored after training? A: No. Model weights are publicly accessible. Anyone can download and run inference locally. There is no central API to censor.

Q: Who controls the project? A: During the Genesis phase, the founder launches and operates the network. As the miner community and infrastructure operators join, governance authority is progressively transferred through the tripartite checks and balances mechanism. The ultimate goal is that no individual or entity holds non-shareable control over Alice.

Q: Why 21 million supply? A: Identical to Bitcoin's total supply. Recognizable, credible, clear scarcity narrative.

Conclusion

Alice represents the first serious attempt to train artificial intelligence through pure decentralization — not pretrained on corporate models, no venture capital, no company.

This is not a business. This is a protocol — a set of open rules that anyone can run.

If it succeeds, Alice will prove that: AI training can be decentralized; intelligence need not be monopolized; cognitive sovereignty can be achieved.

The stakes are existential: if a few companies control AI, they control the future. Alice is the alternative.

Join us.

Source Code: MIT License, open-source **Address Prefix**: SS58 = 300 ('a' prefix)

This whitepaper is version 1.0. Updated March 2026.