Hanzo DSO: Decentralized Semantic Optimization with Byzantine-Robust Prior Aggregation

Hanzo Industries Inc.
995 Market St, San Francisco, CA
contact@hanzo.ai

October 2025

Abstract

We present **Hanzo DSO** (Decentralized Semantic Optimization), a protocol for sharing and aggregating experiential priors across distributed language model agents without parameter updates. DSO enables zero-training model adaptation by (i) extracting semantic advantages via local TF-GRPO, (ii) compressing priors with 1-bit quantization ($29.5\times$ savings), (iii) aggregating contributions via byzantine-robust median voting, and (iv) synchronizing via IPFS/Arweave with on-chain registry verification. Key contributions: (a) Byzantine-robust aggregation scheme with stake-weighted quality scores, (b) Content-addressed experience registry with Merkle proofs, (c) Gossip-based P2P sync optimized for high-quality shard propagation, and (d) Integration with Hanzo Network's PoAI consensus for attestation-based rewards. We demonstrate **15.2% improvement** in multi-agent code generation tasks when nodes share priors vs. isolated operation.

1 Introduction

Traditional federated learning aggregates gradients or parameters, requiring compute-intensive operations and risking catastrophic interference. Recent work on parameter-efficient adaptation (LoRA, adapters) reduces overhead but still requires training. In-context learning enables zero-shot adaptation but lacks systematic mechanisms for sharing knowledge across agents.

Our Contribution. Hanzo DSO introduces a decentralized protocol for sharing *experiential priors*—compressed semantic advantages extracted from agent rollouts—enabling collective intelligence without parameter updates.

2 System Architecture

2.1 Roles and Components

- **Agents:** Run local ASO (TF-GRPO), extract and compress priors
- **Registry:** On-chain smart contract storing CIDs, Merkle roots, quality scores
- **Storage:** Off-chain IPFS/Arweave for prior data
- **Aggregators:** Compute byzantine-robust median under fixed schema
- **Validators:** Verify attestations via PoAI, slash malicious submissions

3 Decentralized Semantic Optimization (DSO)

3.1 Experience Priors

Each agent/node maintains an *experience prior* E: token/embedding-level memory distilled from rollouts. Locally, nodes run **Active Semantic Optimization (ASO)** to extract *semantic advantages* from groups of rollouts (TF-GRPO). Priors are compressed (§5) and written to the on-chain *ExperienceRegistry* with Merkle proofs.

3.2 Training-Free GRPO as Bayesian PoE

For a base model with conditional $p_\theta(y \mid x)$ and a set of experiences $\{e_k\}$ mapping to token-level factors $\phi_k(y \mid x)$, decoding uses a *product-of-experts*:

$$p(y \mid x, E) \propto p_\theta(y \mid x) \prod_k \phi_k(y \mid x)^{\alpha_k}, \quad \alpha_k \geq 0. \tag{1}$$

Here ϕ_k are distilled from group-relative semantic advantage; weights α_k are learned by introspective calibration without gradient updates to θ.

3.3 Distributed Aggregation

Hanzo Network aggregates *priors, not gradients.* Let node priors be $\{E_i\}$. We publish hashes and quality scores; the chain computes a byzantine-robust aggregate $\tilde{E} = \text{median}_q\{E_i\}$ under a fixed schema (token bins / embedding centroids). Conflicting contributions resolve by stake-weighted quorum plus quality caps.

4 ExperienceRegistry and P2P Sync

Registry. On-chain contract stores: content-addressed CID, Merkle root, schema version, quality vector, submitter, slashing bond. **Storage.** Off-chain IPFS/Arweave; local SQLite+LanceDB with Merkle verification. **Sync.** Gossip protocol with CRDT merge; priority given to high-quality shards (fee rebates bias peers to propagate them).

5 1-Bit Semantic Compression

Inspired by BitDelta, we store only the *signs* of per-bucket deltas plus per-matrix scales. For an experience matrix $\Delta \in \mathbb{R}^{n \times m}$,

$$\widehat{\Delta} = \alpha \ \text{Sign}(\Delta), \quad \alpha = \tfrac{1}{nm} \sum_{ij} |\Delta_{ij}|. \tag{2}$$

Scales are distilled by matching logits to a teacher rollout. We observe $\approx 29.5\times$ storage savings with negligible loss in downstream utility, enabling multi-tenant caching and rapid hot-swaps of personalizations.

5.1 Compression Algorithm

Algorithm 1 BitDelta-Inspired Compression

```
input: full-precision experience matrix $\Delta \in \mathbb{R}^{n\times m}$
Compute average absolute value: $\alpha = \frac{1}{nm}\sum_{ij} |\Delta_{ij}|$
Quantize: $\widehat{\Delta}_{ij} = \alpha \cdot \text{sign}(\Delta_{ij})$
Store: binary signs $\{\text{sign}(\Delta_{ij})\}$ and scalar $\alpha$
return compressed representation: $(\{\text{sign}(\Delta_{ij})\}, \alpha)$
```

5.2 Decompression and Application

At inference time:

$$\Delta_{ij}^{\text{approx}} = \alpha \cdot \text{sign}(\Delta_{ij}), \tag{3}$$

yielding a 1-bit per element representation plus one scalar per matrix block. This enables efficient storage and rapid loading of experience priors.

6 Byzantine-Robust Aggregation

6.1 Threat Model

Adversaries may submit:

- **Random noise:** Low-quality priors to pollute aggregate
- **Targeted attacks:** Priors designed to degrade specific tasks
- **Sybil attacks:** Multiple identities voting for malicious priors

6.2 Median Voting Under Schema

Fix a schema $\mathcal{S}$: token bins or embedding centroids. Each submission E_i is decomposed:

$$E_i = \{\Delta_i^{(s)}\}_{s\in\mathcal{S}}, \tag{4}$$

where $\Delta_i^{(s)}$ is the advantage for schema element s. The aggregate computes:

$$\tilde{E}^{(s)} = \text{weighted-median}_i(\Delta_i^{(s)}; w_i), \tag{5}$$

with weights w_i proportional to stake times quality score q_i.

6.3 Quality Scoring

Quality scores $q_i \in [0, 1]$ are estimated via:

1. **Holdout validation:** Test priors on reserved benchmarks
2. **Cross-validation:** Peer nodes sample and verify
3. **Reputation:** Historical performance on-chain

Quality below threshold $q_{\min}$ (default 0.3) triggers bond slashing.

7 ExperienceRegistry Contract

7.1 Interface

```
interface IExperienceRegistry {
  struct Entry {
    bytes32 merkleRoot;
    string cid;          // IPFS/Arweave
    uint64  schema;
    uint64  quality;     // quantized [0, 1000]
    address submitter;
    uint256 bond;
  }
  function submit(Entry calldata e) external payable
    returns (uint256 id);
  function voteQuality(uint256 id, uint64 score) external;
  function slash(uint256 id, address challenger,
    bytes calldata proof) external;
  function aggregate(uint256[] calldata ids) external
    returns (bytes32 aggregateMerkleRoot);
}
```

7.2 Bond and Slashing

Submission requires bond D (default 25 \$AI). If $q_i < q_{\min}$, challenger submits proof (counterexamples); successful challenge burns σD (default $\sigma = 0.5$), refunds $(1 - \sigma)D$ to challenger.

8 P2P Synchronization

8.1 Gossip Protocol

Nodes maintain local replica of high-quality priors:

1. Subscribe to registry events (new submissions, quality updates)
2. Fetch CID from IPFS/Arweave if $q_i \geq q_{\text{fetch}}$ (default 0.5)
3. Verify Merkle proof against on-chain root
4. Merge into local LanceDB with CRDT semantics

8.2 Incentive Alignment

Nodes that propagate high-quality priors earn fee rebates:

$$\text{rebate}_i \propto \sum_{j:\text{fetched from } i} q_j \cdot \text{size}_j. \tag{6}$$

9 Integration with PoAI

9.1 Attestation-Based Rewards

Each prior submission includes PoAI attestation (TEE report + task metrics):

- ΔI: Information gain from experience
- ΔU: Utility improvement on held-out tasks
- Cost metrics: compute, bandwidth, energy

Emissions formula (see HMM paper):

$$R_i = \gamma \Delta I + \beta \Delta U - \lambda_c \cdot \text{cost}_i, \tag{7}$$

where $\gamma = 1.0$, $\beta = 0.5$, $\lambda_c = 0.1$ (defaults).

10 Experimental Evaluation

10.1 Multi-Agent Code Generation

Configuration	Resolved Rate	Avg. Prior Reuse
Isolated agents (no DSO)	16.3%	0.0
DSO (Byzantine honest)	18.8%	3.2 priors/task
DSO (20% Byzantine)	17.9%	2.8 priors/task
DSO (median voting)	**18.7%**	**3.1 priors/task**

Table 1: SWE-bench Verified (500 issues), 10 agents. Byzantine nodes submit random priors.

10.2 Storage Efficiency

- Full-precision priors: 2.4 GB/agent
- 1-bit compressed: 82 MB/agent (29.3× savings)
- IPFS overhead: +12% (metadata, proofs)
- Effective: ≈ 26× savings

11 Related Work

Federated learning: FedAvg, FedProx, byzantine-robust aggregation. **Decentralized ML:** Swarm learning, peer-to-peer training. **Parameter-efficient adaptation:** LoRA, adapters, prompt tuning. **Blockchain + ML:** Federated learning on blockchain, decentralized model markets.

12 Conclusion

Hanzo DSO enables decentralized, zero-training model adaptation by sharing compressed experiential priors. Byzantine-robust aggregation ensures resilience against adversarial nodes, while integration with PoAI provides attestation-based incentives. Future work includes cross-domain transfer (code → data science) and hierarchical aggregation (specialized sub-groups).

A Security Analysis

A.1 Sybil Resistance

Stake-weighting limits influence of Sybil identities. With N honest nodes and $M < N/2$ Sybil identities (each with stake s), median voting ensures honest aggregate if total honest stake $\geq$ total malicious stake.

A.2 Data Poisoning

1-bit quantization limits information content per prior, reducing attack surface. Median voting filters extreme values. Quality scoring enables post-hoc detection.

Disclaimer. This document describes a proposed protocol. Security properties require formal verification.