NeuroSpace: A Protocol for Verifiable and Traceable Language Model Inference

Anonymous
https://neurospace.chat

May 7, 2025

Abstract

Large language models (LLMs) are increasingly central to communication, decision-making, and computation. Yet their outputs remain unverifiable, mutable, and tied to opaque infrastructure. NeuroSpace introduces a decentralized protocol for verifiable machine intelligence. It combines cryptographic hashing, decentralized storage (IPFS), and smart contract commitments to produce transparent, traceable, and tamper-evident AI outputs.

Through on-chain attestations and wallet-based signatures, NeuroSpace allows any observer to independently audit a model's response, its input, its metadata, and its provenance. The protocol supports verifiable chat sessions, Retrieval-Augmented Generation (RAG), and the future of traceable autonomous agents. A native utility token, NeuroCoin (NSPACE), aligns incentives and powers usage across the platform.

NeuroSpace is not merely a tool for interaction—it is a foundation for accountable artificial intelligence.

1 Introduction

Large language models (LLMs) have transformed how information is generated, queried, and consumed. From summarizing documents to answering technical questions, they now mediate vast amounts of digital cognition. Yet despite their power, modern LLMs are opaque: their outputs are unverifiable, their behavior untraceable, and their deployment largely centralized.

This lack of transparency presents both technical and ethical challenges. How can users trust a model's answer? How can third parties audit what was said, when, and why? And how can we attribute responsibility for critical decisions made by algorithmic agents?

NeuroSpace addresses these challenges by introducing a cryptographically secure protocol for verifiable machine intelligence. At its core, NeuroSpace transforms each LLM interaction into a formally verifiable event. Prompt, response, and metadata are hashed, signed, and pinned to decentralized storage. These cryptographic commitments are then submitted to a smart contract, producing a permanent, auditable record on-chain.

This architecture enables users and third parties to:

- Reconstruct and verify the integrity of any message
- Attribute outputs to specific wallets or agents
- Audit entire chat sessions or agent execution traces
- Ground model outputs in immutable external documents (RAG)

In parallel, NeuroSpace introduces a native token, **NeuroCoin (NSPACE)**, to power its economic and governance layers. The token serves as the medium for payments, staking, and future coordination.

Together, these components form a new substrate for trusted AI infrastructure—one in which language model outputs are no longer ephemeral, but permanent, verifiable, and provably accountable.

2 Technical Protocol Overview

2.1 Verifiability Layer

The Verifiability Layer in NeuroSpace ensures that each interaction with a language model is publicly auditable and cryptographically secured. This section defines the interaction structure, commitment scheme, and verification protocol.

Interaction Model

Each user interaction is modeled as a structured tuple:

$$m = (\text{prompt}, \text{response}, \text{metadata}) \in \mathcal{M}$$

where:

- `prompt`: The user-provided natural language input.
- `response`: The LLM-generated output.
- `metadata`: Model configuration (name, temperature, token limit), session ID, timestamp t, and wallet address w.

Hash Commitment

A cryptographic commitment is computed using SHA-256:

$$h = H(\text{Serialize}(m)), \quad H : \{0,1\}^* \to \{0,1\}^{256}$$

where `Serialize` is a deterministic, bijective encoding function over $\mathcal{M}$. This ensures:

$$\text{Deserialize}(\text{Serialize}(m)) = m$$

Thus, any honest verifier can reconstruct h from the published message.

Digital Signature

Let $\text{Sign}_{sk}(h)$ denote an EIP-191-compatible digital signature using a private key sk. The user computes:

$$\sigma = \text{Sign}_{sk}(h)$$

This binds the message to a unique Ethereum address, enabling cryptographic attribution. The corresponding wallet address w can be recovered using:

$$w = \text{RecoverAddress}(h, \sigma)$$

On-Chain Commitment

A smart contract $\mathcal{C}$ maintains a public, append-only ledger of commitments:

$$(h, w, t_{\text{block}}) \in \mathbb{H} \times \mathbb{W} \times \mathbb{T}$$

where:

- h: SHA-256 commitment of the interaction.
- w: Submitting wallet address.
- t_{block}: Timestamp of the block when stored.

Verification Procedure

To verify a message m with signature σ and claimed signer w, the verifier performs:

1. Compute $h' = H(\text{Serialize}(m))$
2. Recover $w' = \text{RecoverAddress}(h', \sigma)$
3. Confirm $w' = w$
4. Check that $(h', w, t) \in \mathcal{C}$

If all checks pass, the interaction is deemed *verifiably committed*.

Security Guarantees

Assuming the collision resistance of SHA-256 and unforgeability of ECDSA, the protocol guarantees:

- **Integrity**: Adversaries cannot forge alternate $m' \neq m$ with $H(m') = H(m)$.
- **Non-repudiation**: Signers cannot deny authorship of valid σ.
- **Immutability**: On-chain records cannot be retroactively modified or deleted.

2.2 Storage and Retrieval

In NeuroSpace, content persistence is achieved through decentralized file storage, ensuring that every prompt-response pair is content-addressable, censorship-resistant, and retrievable for independent verification. The system utilizes the InterPlanetary File System (IPFS) to store serialized message data.

IPFS-Based Storage Model

Given a structured message tuple:

$$m = (\text{prompt}, \text{response}, \text{metadata})$$

the backend serializes m deterministically into a byte array $b \in \{0,1\}^*$. This serialized payload is uploaded to IPFS, resulting in a unique content identifier (CID):

$$\text{CID}_m = \text{IPFS.Put}(b)$$

The CID is a multihash representing the SHA-256 digest of the content and serves as an immutable, globally addressable pointer.

Content-Addressable Integrity

Let:

$$h = H(\text{Serialize}(m))$$

Then under the default IPFS hashing function:

$$\text{CID}_m \equiv \text{Multihash}(h)$$

Any deviation in the content m results in a different CID, preserving strong integrity guarantees. The CID is recorded in the metadata and is optionally stored on-chain alongside the commitment hash for additional auditability.

Metadata Schema

Each stored message is bundled with metadata fields to allow full contextual reconstruction and reproducibility:

- **Model configuration:** name, ID, temperature, max tokens
- **Timestamps:** local timestamp t, block timestamp t_{block}
- **Identity:** wallet address w, session UUID s
- **Verification:** hash h, signature σ, IPFS CID

This metadata is used by verifiers to ensure consistent hash recomputation and signature validation.

Retrieval and Verification Flow

On retrieval, the client performs the following:

1. Fetch raw content via IPFS:
$$b = \text{IPFS.Get}(\text{CID}_m)$$
2. Deserialize to obtain m
3. Compute $h' = H(\text{Serialize}(m))$
4. Verify digital signature σ and wallet w
5. Check on-chain presence of $(h', w) \in \mathcal{C}$

This round-trip provides a full verification circuit from input to content to on-chain proof.

Availability Considerations

IPFS provides decentralized distribution but not persistence guarantees. NeuroSpace addresses this through:

- Redundant pinning across independent pinning services
- Optional integration with Filecoin or Arweave (future work)
- Frontend caching and fallback storage

The design ensures long-term data retrievability without relying on centralized infrastructure.

Security Properties

- **Tamper resistance:** IPFS CID is bound to content hash.
- **Censorship resistance:** Distributed peer-to-peer network.
- **Transparency:** Anyone can resolve CID and verify integrity.

2.3 Session Identity and Traceability

To support structured, longitudinal interactions, NeuroSpace introduces the notion of verifiable sessions. Each session represents a temporally ordered sequence of user-model interactions, cryptographically bound to a user identity and persistently linked across messages.

Session Definition

Let a session be defined as:

$$S = (s, w, \{m_1, m_2, \ldots, m_n\})$$

where:

- $s \in \mathcal{U}$: A globally unique session identifier (UUID).
- $w \in \mathbb{W}$: Ethereum wallet address initiating the session.
- $m_i \in \mathcal{M}$: Ordered messages generated within the session.

Each message m_i is defined as in Section 2.1 and is independently verifiable.

Session Binding

All messages within a session share the same session ID s and wallet address w. The metadata for each message contains:

$$\texttt{metadata} = \{\ldots, s, w, t_i\}$$

This allows for:

- Temporal reconstruction of the session history.
- Identification of the originating wallet.
- Hash and signature consistency checks across session scope.

Deterministic Lineage and Replay

Let $\mathcal{H}_S$ be the ordered set of committed message hashes in a session:

$$\mathcal{H}_S = [h_1, h_2, \ldots, h_n] \quad \text{where } h_i = H(\text{Serialize}(m_i))$$

This sequence represents the immutable record of the session. Assuming the LLM is deterministic given fixed configuration, the session can be replayed or independently simulated.

Traceability Guarantees

By design, the session structure provides the following guarantees:

- **Causal ordering:** Each message is timestamped and ordered by submission time.
- **Attribution:** All messages are signed by the same wallet address w.
- **Replayability:** Full session reconstruction is possible given stored messages and metadata.
- **Non-equivocation:** Forking or altering session history requires a hash/signature mismatch.

Applications of Traceability

The verifiable session structure supports higher-order use cases:

- Auditable multi-turn conversations
- Agent memory logs with verified steps
- Cross-session reputation systems
- Forensic analysis and provenance tracking

In the future, sessions may be extended with external tools, agent state transitions, or encrypted embeddings, further enriching their traceability potential.

2.4 Verifiable Retrieval-Augmented Generation (RAG)

Retrieval-Augmented Generation (RAG) enhances LLM performance by incorporating external documents into the generation process. In NeuroSpace, we extend RAG with cryptographic verifiability, ensuring that all cited sources are immutable, content-addressed, and independently inspectable.

RAG Formal Model

Let:

- q: A user query (prompt)
- $D = \{d_1, d_2, \ldots, d_n\}$: A retrieved document set
- $r = \text{LLM}(q, D)$: The model-generated response

Each document $d_i \in D$ is stored on IPFS and associated with a unique content identifier:

$$h_i = H(d_i) = \text{SHA-256}(d_i) \quad \text{and} \quad \text{CID}_i = \text{Multihash}(h_i)$$

The set of document hashes used in a RAG call is denoted:

$$\mathcal{H}_D = \{h_1, h_2, \ldots, h_n\}$$

Binding Sources to Response

The response r is generated with explicit dependence on D. NeuroSpace binds r to the retrieval set by extending the verification payload:

$$m = (q, r, \mathcal{H}_D, \text{metadata})$$

This tuple is serialized and hashed as in Section 2.1:

$$h = H(\text{Serialize}(m))$$

The resulting hash is signed and submitted on-chain. The associated IPFS upload includes:

- The raw documents $\{d_1, \ldots, d_n\}$
- The computed response r
- The query q
- Verification metadata

Verification Workflow

To verify a RAG response, a client must:

1. Retrieve and hash each source document to confirm $h_i = H(d_i)$
2. Confirm each $h_i \in \mathcal{H}_D$
3. Recompute the verification hash from $(q, r, \mathcal{H}_D, \text{metadata})$
4. Verify the signature and on-chain hash commitment

This ensures that every cited source used in generation is publicly auditable, fixed at the time of response, and resistant to tampering.

RAG Security Guarantees

Under standard hash assumptions:

- **Source integrity**: No document can be swapped or altered without breaking the hash.
- **Provenance transparency**: Every fact in r is traceable to a specific document in D.
- **Tamper detection**: Any modification to D or r is detectable by recomputation.

Applications of Verifiable RAG

- Fact-checkable AI answers with cited, immutable sources
- Regulatory compliance and audit trails for enterprise AI
- Trusted medical, legal, and scientific assistance
- Verifiable customer support or document summarization

Future versions of the protocol may integrate document ranking scores, embeddings, and source attribution weights, all stored verifiably as part of the RAG context.

2.5 Agent Extensions (Future Work)

While NeuroSpace currently supports verifiable single-turn interactions and session-level traceability, future protocol extensions will enable verifiable autonomous agents. These agents will perform multi-step reasoning and tool use, with each internal action and output logged as part of a cryptographically traceable execution graph.

Agent Execution Model

Let an agent $\mathcal{A}$ be defined as a stateful process:

$$\mathcal{A} : \Sigma \to \Sigma$$

where Σ represents the agent's internal state, including memory, tools, and contextual history.

An execution trace of the agent is modeled as a sequence of steps:

$$T = [\tau_1, \tau_2, \ldots, \tau_k]$$

Each step τ_i represents a single action and has the form:

$$\tau_i = (\text{input}_i, \text{action}_i, \text{output}_i, \text{metadata}_i)$$

The metadata may include:

- Timestamp of execution
- Tool or function invoked
- Model configuration
- External context or API results

Trace Hash Commitments

Each step τ_i is serialized and hashed:

$$h_i = H(\text{Serialize}(\tau_i))$$

The entire trace is recorded as a directed acyclic graph (DAG), where edges represent causal or functional dependencies between steps. A final root hash h^* can be computed via Merkleization or hash chaining to produce a succinct, verifiable commitment:

$$h^* = \text{RootHash}(T)$$

This root hash may be submitted on-chain and signed by the agent's controlling wallet, or in some cases by a validator node or verifier.

Verification and Auditing

Any third party can retrieve the agent trace from IPFS, verify:

- Internal consistency of the steps (hash chain or Merkle path)
- Signature over the root hash
- On-chain existence of the commitment

Given that each step is tied to a specific model invocation, tool call, or environment interaction, this enables full transparency into agent behavior.

Use Cases for Verifiable Agents

- Auditable reasoning paths for complex questions
- Regulatory transparency in AI-assisted decision-making
- On-chain market of agent tasks with trust guarantees
- Agent-specific performance tracking and reputation scores

Design Considerations

Future versions of the protocol will explore:

- Lightweight proof systems for agent state transitions
- Zero-knowledge variants for privacy-preserving traces
- Staking mechanisms for agent honesty
- Incentives for third-party validators to audit traces

These extensions aim to generalize the NeuroSpace protocol from verifiable messages to verifiable computation, enabling composable and accountable AI agents.

3 Decentralized Model Feedback and Performance Alignment

A critical challenge in AI system design is collecting reliable feedback on model quality, safety, and usefulness. Most existing systems rely on centralized, opaque mechanisms controlled by the model provider. NeuroSpace replaces this paradigm with a decentralized protocol for open model evaluation, aligning user feedback with verifiable signals and economic incentives.

Feedback as Verifiable Data

Users interacting with a model on NeuroSpace may submit structured feedback on any response:

$$f = (\text{message\_id}, \text{feedback\_type}, \text{note}, w)$$

where:

- `message_id`: Reference to the model output under review
- `feedback_type` $\in$ {hallucination, inaccuracy, harm, bias, other}
- `note`: Optional free-form explanation
- w: Wallet address of the submitting user

All feedback is signed, timestamped, and stored either on-chain or within verifiable off-chain logs. This produces a decentralized, tamper-resistant corpus of model critiques.

From Feedback to Insight

This feedback layer enables granular analysis of model performance:

- Identify frequently flagged failure modes
- Monitor model versions over time
- Compare models across dimensions of trustworthiness, bias, or hallucination frequency
- Generate structured datasets for supervised fine-tuning or RLHF pipelines

Rather than outsourcing model improvement to a closed internal team, NeuroSpace exposes this process to the broader community.

Incentive-Aligned Voting and Staking

To reinforce feedback quality, users may stake NSPACE to support or contest the feedback of others. This creates a signaling mechanism for validating which flags are credible and which may be malicious or inaccurate. This mechanism promotes:

- Peer review of feedback quality
- Economic alignment for honest reporting
- Community consensus on model strengths and weaknesses

Validated feedback increases a user's reputation score and may yield rewards. Conversely, consistent bad-faith behavior may result in slashing or decreased credibility.

Decentralized Judgment in the Age of AI

Crucially, NeuroSpace removes the power of moderation and model evaluation from any single entity. Instead, community consensus—backed by transparent records and aligned incentives—determines what constitutes quality output.

This is essential for building a protocol-layer AI system in which feedback is not just data but governance. It opens the door to:

- Publicly maintained model leaderboards
- Reputation systems for model instances and providers
- Community-weighted tuning of content filters or system prompts

Future Directions

The decentralized feedback framework can be extended to support:

- Scalar ratings on helpfulness, informativeness, and clarity
- Feedback-linked performance dashboards for model developers
- Reputation-weighted voting schemes
- Feedback-conditioned fine-tuning or model routing

NeuroSpace turns feedback into a core protocol primitive—shifting model optimization from private labs to public consensus.

4 Token Economics: NeuroCoin (NSPACE)

NeuroCoin (symbol: **NSPACE**) is the native utility token of the NeuroSpace platform. While initial deployments support Ethereum-based pay-per-query payments, the long-term design centers NSPACE as the economic and governance backbone of the protocol.

4.1 Token Parameters

- **Symbol:** NSPACE
- **Total Supply:** 100,000,000 tokens (fixed)
- **Decimal Precision:** 18
- **Network:** Ethereum-compatible (Base L2 mainnet)

4.2 Utility Functions

NSPACE facilitates secure, programmable, and incentive-aligned interaction with the protocol. Core utility includes:

- **Query Payments:** End users pay in NSPACE to access LLM inference or invoke agent actions.
- **Staking:** Token holders may stake NSPACE to:
  - Prioritize computation requests
  - Guarantee agent execution integrity (future work)
  - Back storage and verification nodes
- **Governance:** NSPACE may be used to vote on protocol parameters, supported models, storage policies, and incentive structures.

4.3 Token Allocation (Tentative)

To bootstrap protocol utility and growth, the initial allocation of NSPACE may follow:

- 40% – Community and Ecosystem Incentives
- 25% – Founding Team and Core Developers (vesting over 3-4 years)
- 20% – Contributors and Advisors
- 15% – Protocol Treasury and Grants

4.4 Economic Design Principles

The NSPACE token adheres to the following design principles:

1. **Sustainability:** Token payments are used to fund verifiers, pinning nodes, and compute infrastructure.
2. **Decentralization:** No centralized authority controls the token supply or protocol evolution after launch.
3. **Incentive Alignment:** All stakeholders—users, model providers, validators, and auditors—are economically incentivized to act honestly.

4.5 Token Lifecycle and Circulation

- **Initial Phase:** ETH-based pay-per-query (current MVP) with soft transition to dual ETH/NSPACE support.
- **Adoption Phase:** Native NSPACE payments required for most core functionality (staking, priority execution, RAG extensions).
- **Maturity Phase:** Governance, model reputation markets, and protocol grants administered entirely in NSPACE.

4.6 Future Considerations

Future versions of the protocol may explore:

- Dynamic pricing mechanisms for token-based model access
- Burn-and-mint cycles tied to compute usage
- ZK-commitments for private token-based queries
- Cross-chain support for broader token composability

5 Conclusion

NeuroSpace introduces a protocol for verifiable and traceable machine intelligence. By combining cryptographic commitments, decentralized storage, and on-chain attestations, it transforms black-box LLM interactions into transparent, auditable computational artifacts.

Through the layered architecture of prompt hashing, digital signature binding, IPFS-based content addressing, and smart contract verification, NeuroSpace establishes the infrastructure for publicly verifiable AI inference. Its support for session tracking, document-grounded generation, and future agent-level traceability extends this integrity across increasingly complex AI workflows.

The NeuroCoin (NSPACE) token powers this ecosystem by aligning incentives across users, model providers, and protocol validators. NSPACE enables permissionless participation in the

platform's economic and governance layers, anchoring its long-term decentralization and sustainability.

What has been built is more than a chat application—it is a live demonstration of Vitalik Buterin's original vision for decentralized applications: transparent, trust-minimized, and credibly neutral. NeuroSpace is not merely a platform; it is a foundation for a new standard of machine intelligence—one in which outputs are not just plausible, but provable.

NeuroSpace: Verifiable. Traceable. Decentralized Intelligence.