Semantic Event Protocol (SEP)

Event-Driven Distributed Intelligence

Whitepaper v2.0 — December 2025

Nikolay Yudin
1@seprotocol.ai

seprotocol.ai

Abstract

The Semantic Event Protocol (SEP) is an experimental framework for event-driven distributed computing. Instead of continuous data transmission, nodes communicate only when semantic state changes exceed a defined threshold. This document presents the protocol's design principles and preliminary experimental results from small-scale controlled tests.

Preliminary Findings (single author, awaiting replication):

- **32× bandwidth reduction** in distributed training via ternary quantization (17MB → 271KB per sync)
- **93% transfer efficiency** between different model architectures (DistilBERT → GPT-2)
- **100% compositional generalization** with HDC on toy tasks (vs. 21% for Transformers)

Important limitations: These results come from controlled benchmarks by a single researcher. They suggest promising directions but require independent validation before any production consideration. We publish this work to invite scrutiny and collaboration.

Contents

Introduction

Motivation

Modern AI development is characterized by increasing centralization. Training frontier models requires capital investments exceeding $100M, with the majority allocated to GPU compute. Hardware access is further constrained by export controls and supply chain dependencies.

This creates a structural challenge: entities without datacenter-scale resources have limited paths to AI capability development.

Research Question

Can distributed, event-driven architectures provide a viable alternative path for AI development? Specifically:

1. Can bandwidth requirements for distributed training be reduced to edge-viable levels?
2. Can knowledge transfer occur between different model architectures?
3. Can compositional reasoning be achieved through non-Transformer approaches?

Scope and Limitations

This whitepaper presents:

- A protocol specification for semantic event communication
- Preliminary experimental results from controlled benchmarks
- Open questions and areas requiring further research

Note on Evidence Quality: All experiments described here were conducted by a single author using synthetic benchmarks and small-scale tests. Results should be interpreted as preliminary findings that suggest directions for further investigation, not as production-ready solutions.

Background

The Economics of AI Training

Training costs for frontier models have grown exponentially:

Table 1: Estimated Training Cost Structure

Component	Share	Estimated Cost
GPU Compute	60-70%	$60-70M
Electricity/Cooling	10%	$10M
Data & Labeling	5-10%	$5-10M
Personnel	10-15%	$10-15M
Infrastructure	5%	$5M
Total	100%	**$100M+**

The GPU compute component represents a critical bottleneck, as it requires both capital and hardware access that may be restricted.

Existing Approaches to Distributed Training

Several projects have explored distributed training:

- **Hivemind** [3]: Collaborative training across heterogeneous hardware
- **DiLoCo** [4]: Distributed low-communication training
- **BitNet** [5]: Ternary quantization for efficient inference

These approaches demonstrate feasibility but face challenges in bandwidth, synchronization, and cross-architecture compatibility.

Hyperdimensional Computing

Hyperdimensional Computing (HDC) uses high-dimensional vectors (typically 10,000 dimensions) with algebraic operations:

- **Binding ($\otimes$):** Associative operation where $A \otimes B$ is dissimilar to both A and B
- **Bundling ($+$):** Set-like operation where $A + B$ is similar to both A and B
- **Permutation (ρ):** Positional encoding for sequences

HDC has been explored for edge inference due to its noise tolerance and computational simplicity [2].

The Semantic Event Protocol

Core Principle

SEP is based on one axiom:

Nodes compute and communicate only when semantic state changes exceed a threshold.

This contrasts with clock-driven systems where computation occurs at fixed intervals regardless of information value.

Protocol Design

3.2.1 Semantic Events

The fundamental communication unit is the Semantic Event:

$$E = (\text{context}, \Delta\mu, \text{confidence}, \text{provenance})$$

Where $\Delta\mu$ represents the change in semantic state, encoded as a vector delta.

3.2.2 Transmission Condition

A node transmits when:

$$d(M_t, M_{t-1}) > \theta$$

Where d is a distance function (e.g., cosine distance) and θ is a configurable threshold.

3.2.3 Semantic Alignment

Different nodes may use different embedding models. SEP uses Orthogonal Procrustes alignment:

```
R = orthogonal_procrustes(anchors_A, anchors_B)
v_aligned = v_foreign @ R
```

This allows nodes with different local models to communicate in a shared semantic space.

Protocol Stack

L5: Collective Cognition — Emergent mesh behavior
L4: Semantic Sharing — Gossip protocol
L3: Compression — HDC + Ternary quantization
L2: Local Cognition — Node-level processing
L1: Local Semantics — Embedding extraction
L0: Sensory Input — Raw data processing

Figure 1: SEP Protocol Stack

Experimental Results

This section presents preliminary experimental findings. All experiments were conducted by a single researcher using controlled benchmarks. We present these results to invite scrutiny and replication attempts.

Reproducibility: Code for all experiments is available at https://github.com/nick-yudin/SEP. We encourage independent replication.

Experiment 1: HDC Bandwidth Compression (M3b)

4.1.1 Setup

- **Model:** OPT-350m with LoRA adapters
- **Nodes:** 2 (simulated distributed environment)
- **Sync mechanism:** Firebase Realtime Database
- **Compression:** Ternary quantization {-1, 0, +1} with 70% sparsity

4.1.2 Compression Pipeline

1. Flatten LoRA weight tensors
2. Quantize to ternary values with threshold-based sparsity
3. Pack 4 values per byte (2 bits each)
4. Base64 encode for transmission

4.1.3 Results

Metric	Uncompressed	Compressed	Reduction
Bandwidth/round	17.5 MB	271 KB	64×
Compression ratio	1×	32×	—
Final loss	1.92	2.02	+5%

4.1.4 Interpretation

The 32× compression with 5% loss increase suggests that ternary quantization may be viable for distributed training synchronization. However, this was tested on a single small model with only 2 nodes. Scaling behavior remains unknown.

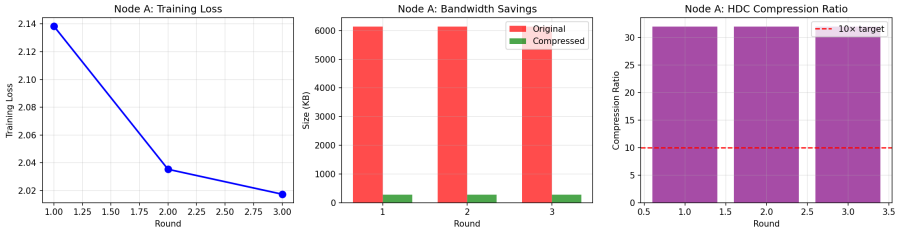

Figure 2: Bandwidth comparison: Original (17.5 MB) vs. Compressed (271 KB)

Experiment 2: Cross-Architecture Transfer (M3c)

4.2.1 Hypothesis

Knowledge learned by one model architecture can transfer to a different architecture via semantic examples, without weight sharing.

4.2.2 Setup

- **Teacher:** DistilBERT (encoder architecture, 66M parameters)
- **Student:** GPT-2 (decoder architecture, 124M parameters)
- **Task:** Sentiment classification (SST-2)
- **Transfer mechanism:** 320 labeled examples with semantic embeddings

4.2.3 Method

1. Train Teacher on 2,000 examples
2. Extract "knowledge packet": examples where Teacher improved + high-confidence predictions
3. Train Student on knowledge packet only (no original training data)
4. Measure Student accuracy

4.2.4 Results

Model	Before Training	After Training
Teacher (DistilBERT)	49.0%	86.6%
Student (GPT-2)	47.0%	82.0%

$$\text{Transfer Efficiency: } \frac{35.0\%}{37.6\%} = 93.1\%$$

4.2.5 Interpretation

The Student achieved 93% of the Teacher's improvement using only the knowledge packet. This suggests that semantic example transfer may enable heterogeneous distributed networks. However:

- This was tested on a simple classification task
- Both models are relatively small
- The task (sentiment) may be particularly amenable to this approach
- Generalization to more complex tasks is untested

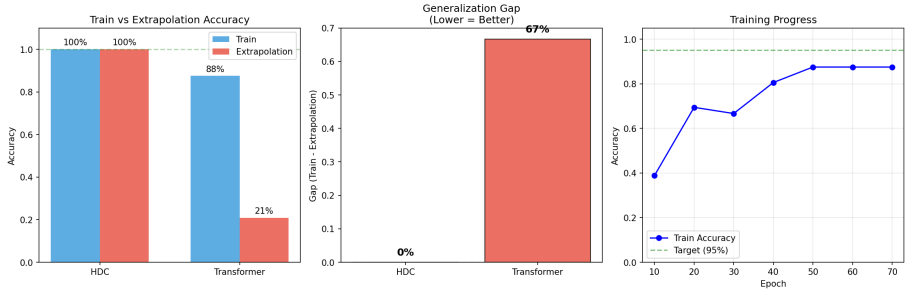

Figure 3: Compositional generalization: HDC 100% vs. Transformer 21%

Experiment 3: Compositional Generalization (M2.6)

4.3.1 Hypothesis

Transformers learn statistical patterns but struggle with compositional generalization to unseen combinations. HDC's algebraic structure should enable perfect generalization.

4.3.2 Setup

- **Task:** Command language interpretation
- **Primitives:** `walk`, `run`, `swim`
- **Modifiers:** `twice`, `four times`
- **Holdout:** `swim four times` never seen during training

4.3.3 Results

Model	Train Accuracy	Extrapolation (Unseen)
HDC	100%	100%
Transformer (1M)	88%	21%
Transformer (31M)	91%	0%

4.3.4 Interpretation

On this toy task, HDC demonstrates perfect compositional generalization while Transformers fail. Notably, scaling the Transformer did not help—the larger model achieved *worse* extrapolation (0% vs 21%).

Important caveat: This is a synthetic, simplified task. Whether HDC's compositional properties transfer to real-world applications is an open question. The result is consistent with prior work on Transformer compositional limitations [6], but should not be over-interpreted.

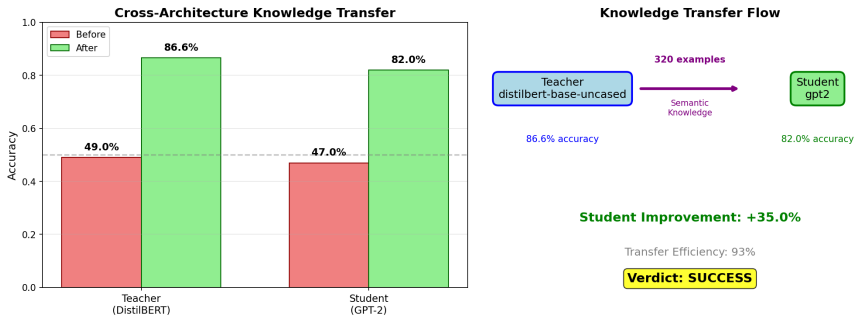

Figure 4: Cross-architecture transfer: Teacher 86.6% → Student 82.0%

Summary of Experimental Status

Table 2: Experimental Evidence Summary

Experiment	Finding	Status	Confidence
M3b: Compression	32× reduction, 5% loss increase	Demonstrated	Medium
M3c: Cross-arch	93% transfer efficiency	Demonstrated	Medium
M2.6: Composition	HDC 100% vs Transformer 21%	Demonstrated	Low-Medium

Confidence levels reflect: single-author experiments, synthetic benchmarks, small scale. Independent replication would significantly increase confidence.

Open Questions

Our preliminary results raise more questions than they answer:

Scaling

- Does 32× compression hold for larger models (7B+)?
- How does cross-architecture transfer scale with task complexity?
- What happens with 10, 100, or 1000 nodes?

Real-World Applicability

- Do these results transfer from synthetic benchmarks to production workloads?
- What failure modes exist that our controlled tests didn't expose?
- What security and trust implications arise in distributed settings?

Hardware Dependencies

- Ternary computing requires hardware support that doesn't exist at scale
- Current results are emulated on standard GPUs
- True efficiency gains await neuromorphic/memristive hardware

Governance

- "No one controls it" is not a governance mechanism
- Distributed systems require coordination for updates, conflict resolution
- Economic incentives for participation are undefined

Roadmap

Completed

- ✓ Protocol specification (Level 0, Level 1)
- ✓ Python reference implementation
- ✓ Preliminary benchmarks (semantic filtering, compression, transfer)

Near-Term (Seeking Collaborators)

- ○ Independent replication of key experiments
- ○ Scaling tests (larger models, more nodes)
- ○ Real-world task evaluation

Medium-Term (Research Directions)

- ○ Hardware prototypes (Jetson, Raspberry Pi mesh)
- ○ Governance mechanism design
- ○ Security analysis of distributed training

Long-Term (Speculative)

- Integration with neuromorphic hardware
- Production-scale distributed training
- Federated model ecosystems

Conclusion

The Semantic Event Protocol represents an experimental approach to distributed AI. Our preliminary results suggest that:

1. Significant bandwidth reduction may be achievable for distributed training
2. Knowledge transfer between different architectures appears feasible
3. HDC offers compositional properties that Transformers lack

However, these findings come from controlled benchmarks by a single researcher. They indicate promising research directions, not production-ready solutions.

We publish this work not as a finished product, but as an invitation to collaboration and scrutiny. If these preliminary results hold under independent replication and scaling, they may contribute to a more distributed future for AI development.

If they don't hold, we will have learned something valuable about the limitations of these approaches.

Silence is the default. Meaning is everything.

References

References

[1] Vaswani, A., et al. (2017). "Attention Is All You Need." *NeurIPS 2017*. arXiv:1706.03762

[2] Kanerva, P. (2009). "Hyperdimensional Computing: An Introduction to Computing in Distributed Representation." *Cognitive Computation*.

[3] Diskin, M., et al. (2021). "Distributed Deep Learning in Open Collaborations." *NeurIPS 2021*.

[4] Douillard, A., et al. (2024). "DiLoCo: Distributed Low-Communication Training of Language Models." arXiv:2311.08105

[5] Wang, H., et al. (2023). "BitNet: Scaling 1-bit Transformers for Large Language Models." arXiv:2310.11453

[6] Chollet, F. (2019). "On the Measure of Intelligence." arXiv:1911.01547

How to Cite

```
@misc{sep2025,
  title={Semantic Event Protocol (SEP):
         Event-Driven Distributed Intelligence},
  author={Nikolay Yudin},
  year={2025},
  url={https://seprotocol.ai}
}
```

Nikolay Yudin
1@seprotocol.ai
https://seprotocol.ai
https://github.com/nick-yudin/SEP
Twitter: @Nikolay_Yudin_

December 2025