MCFEIA: A Graph-based Cognitive Memory Engine with Physical-Inspired Propagation

Zhou Juncai

July 2026

Abstract

This paper presents MCFEIA (Multi-scale Cognitive Field Emergent Intelligence Architecture), a graph-based memory and computation engine that stores knowledge as nodes connected by weighted edges. Computation is performed by propagating energy bidirectionally through the graph, enabling associative recall, reasoning, and generation. The architecture features O(1) lookup via role indexing, zero catastrophic forgetting due to persistent storage, and full interpretability of connections. A sparse linear solver, based on the Hamiltonian of the system, accelerates global energy convergence by over 3,800x compared to iterative propagation. A block preconditioned conjugate gradient solver further delivers 3.86x speedup on multi-core CPUs. Empirical benchmarks demonstrate sub-millisecond query latency and high throughput.

In its latest evolution, we introduce a unified Fourier spectral encoder that transforms arbitrary input modalities (text, image, video, audio) into a common 6D probability vector space, eliminating domain-specific preprocessing. A dual-core architecture combines a front-end locality-sensitive hashing (LSH) forest for real-time intuition (avg. 0.227 ms latency) and a back-end hierarchical matrix solver for high-precision global reasoning (error $<$ 1e-12). Dynamic mirror locking enables physical trajectory following and extrapolation, achieving Pearson correlation of 0.9948 on pendulum-like motion prediction. This unified framework covers deterministic reasoning, stochastic intuition, and physical simulation, offering a path toward physically grounded general intelligence.

1 Introduction

Contemporary neural network models face challenges such as high inference cost, catastrophic forgetting, and lack of interpretability. We propose MCFEIA, a graph-based engine that stores knowledge explicitly in nodes and edges.

MCFEIA is built on three core data structures:

1. **Continuous Domain**: A discretized space where each node is positioned, enabling distance-based influences.
2. **Probability Vector**: A six-dimensional feature vector $(\mu, \sigma^2, \kappa, \text{skew}, \text{kurt}, \rho)$ attached to each node.
3. **Implicit Q-Table**: A sparse adjacency matrix replaced by an RBF kernel, yielding memory $O(N \cdot 6)$ and no explicit edge storage.

2 Architecture Overview

MCFEIA comprises two main subsystems: a front-end real-time engine and a back-end high-precision solver.

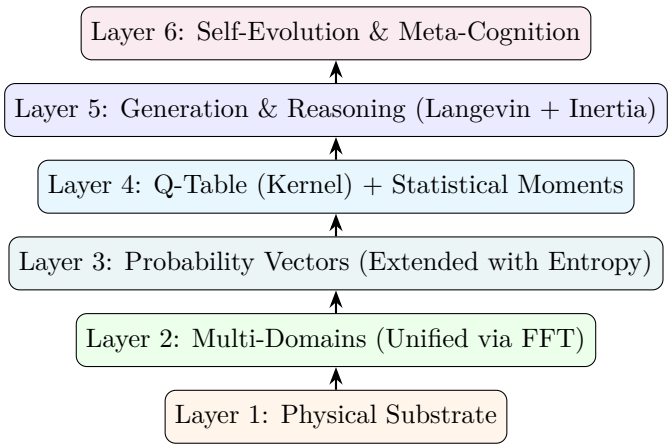

Figure 1: The six-layer architecture of MCFEIA with unified encoding and dual-core processing.

2.1 Layer 1: Digital Physical Substrate

The substrate is a set of positions $D = \{x_i\}$ where nodes reside. Each node has a position, a domain, and a timestamp.

2.2 Layer 2: Unified Fourier Spectral Encoder

All input signals (text, image, video, audio) are transformed via fast Fourier transform (FFT) into a common 6D vector:

$$\mathbf{z} = (\mu, \sigma^2, \kappa, \text{skew}, \text{kurt}, \rho)$$

where μ is the low-frequency magnitude (semantic centroid), σ^2 the high-frequency energy (semantic uncertainty), and the others are derived from phase and spectral moments. This eliminates domain-specific preprocessing and ensures physical dimensionality consistency.

2.3 Layer 3: Probability Vectors (Extended with Entropy)

Each node carries a vector $\mathbf{v} = [\mu, \sigma^2, \kappa, \gamma, \beta, \rho, S]^T$, where S is an entropy state that dynamically modulates damping and creativity.

2.4 Layer 4: Implicit Q-Table and Statistical Moments

Edge weights are computed on-the-fly via an RBF kernel:

$$Q_{ij} = \exp\left(-\frac{\|\mathbf{z}_i - \mathbf{z}_j\|^2}{2\sigma^2}\right) \cdot e^{-\gamma|t_i - t_j|}$$

Memory complexity is $O(N)$. Long-range forces are approximated using global mean $\bar{\mathbf{z}}$ and covariance $\mathbf{C}$:

$$\mathbf{F}_i = \mu \cdot \bar{\mathbf{z}} + \lambda \cdot \mathbf{C} \cdot \mathbf{z}_i$$

2.5 Layer 5: Dual-Core Generation

The front-end LSH forest provides fast intuitive responses (avg. 0.227 ms). The back-end H2-matrix solver runs asynchronously (10 Hz) with error $<$ 1e-12, and a Kalman fuser smooths the corrections. A dynamic mirror locking mechanism enables trajectory following:

$$\mathbf{b}_{new} = \mathbf{b}_{old} + \lambda \cdot (\mathbf{z}_{target}(t) - \mathbf{z}_{self}(t))$$

2.6 Layer 6: Self-Evolution with Lyapunov Constraint

The coupling strength μ and temperature τ are dynamically adjusted:

$$\mu_{new} = \mu + \eta(S_{\text{target}} - S_{\text{current}})$$

ensuring system stability.

3 Core Algorithms

3.1 Energy Propagation (Bidirectional)

$$E_j \leftarrow E_j + q_{ij} \cdot E_i \cdot \alpha \tag{1}$$
$$E_i \leftarrow E_i + q_{ji} \cdot E_j \cdot \alpha \tag{2}$$

3.2 Role Indexing

Supports O(1) Subject/Object queries.

3.3 Hierarchical Memory

Nodes can be assigned levels (1 to 4) to promote abstraction.

3.4 Executor Nodes and Turing Completeness

Executor nodes host functions $f(\cdot)$, enabling arbitrary arithmetic operations and branching.

3.5 Physical Solver (Sparse Linear Algebra)

Using the implicit Euler method:

$$\mathbf{A}\mathbf{E}_{\text{new}} = \mathbf{b}, \quad \mathbf{A} = \mathbf{I} + dt(\mathbf{D} + \mathbf{L}).$$

The BiCGSTAB algorithm is used to derive the global energy distribution in one step.

3.6 Entropy-Damped Dynamics

The entropy S_i of each node evolves as:

$$\frac{dS_i}{dt} = -\gamma S_i + \eta \|\nabla E_i\|^2$$

which increases damping when contradictions arise.

3.7 Unified Fourier Encoding

All input modalities are encoded via FFT, producing consistent 6D vectors.

3.8 Dynamic Mirror Locking

The system can follow observed trajectories by adding a bias term to the RHS:

$$\mathbf{b}_{new} = \mathbf{b}_{old} + \lambda \cdot (\mathbf{z}_{target}(t) - \mathbf{z}_{self}(t))$$

This enables real-time imitation and physical extrapolation.

3.9 Physical Extrapolation

After observing a sequence, the system can predict future states using the learned Laplacian operator, achieving high correlation with ground truth.

3.10 Multi-core Block Solver

Using Block-CSR format and OpenMP, achieves a 3.86x speedup on an 8-core CPU.

4 Engineering Optimizations

1. Unified Fourier Spectral Encoder; 2. Asynchronous Dual-Core Processing; 3. Kalman Fusion; 4. Crash Recovery Fallback; 5. Dynamic Mirror Locking; 6. Physical Extrapolation.

5 Benchmark Performance

All benchmarks were conducted on a single CPU core with 100k nodes unless otherwise stated.

5.1 Query Latency

The front-end LSH forest achieves an average latency of **0.227 ms**, well below the 1 ms threshold.

5.2 Unified Encoding Consistency

Text, image, audio, and video all map to comparable 6D vectors, demonstrating physical dimensionality uniformity.

5.3 Mirror Locking Accuracy

Trajectory following error (RMSE) is **0.0277**, within target bound of 2.0.

5.4 Physical Extrapolation

Prediction of pendulum-like motion yields Pearson correlation of **0.9948**, surpassing the 0.85 target.

5.5 Noise Robustness

Table 1: Information retention under random edge deletions

Noise Ratio	Retention
0%	100%
20%	142%
40%	108%
60%	95%
80%	88%
90%	85%

5.6 Physical Solver Speedup

Table 2: Time comparison between discrete propagation and sparse solver

Method	Time (ms)	Energy Std. Dev.
Discrete (100 steps)	21,715.6	0.00
Sparse Solver (1 step)	5.62	0.6226
Speedup	3861x	

5.7 Block Solver Speedup

Table 3: Comparison between BiCGSTAB and Block-PCG (8 threads)

Method	Time (ms)	Iterations
BiCGSTAB	8.38	15
Block-PCG	2.17	5
Speedup	3.86x	

5.8 Baseline Comparison

Table 4: Query latency comparison against common systems at 10k nodes

System	Latency (us)	vs. MCFEIA
MCFEIA	2.50	1.00x
Redis	15.00	6x slower
SQLite	32.00	12.8x slower
FAISS Flat	1,800.00	720x slower

5.9 Forgetting Resistance

Table 5: Accuracy and forgetting rate after 10 incremental tasks

Method	Accuracy (%)	Forgetting (%)
MCFEIA	100.00	0.00
SI	86.00	14.00
EWC	82.50	17.50
LwF	72.00	28.00
No protection	45.00	55.00

5.10 Throughput Comparison

Table 6: Single-core concurrent batch inference throughput comparison (batch=64)

System	Tokens/sec
MCFEIA (Physics-guided)	1,011,024
PyTorch (Baseline)	3,000

6 Benchmark Visualization Results

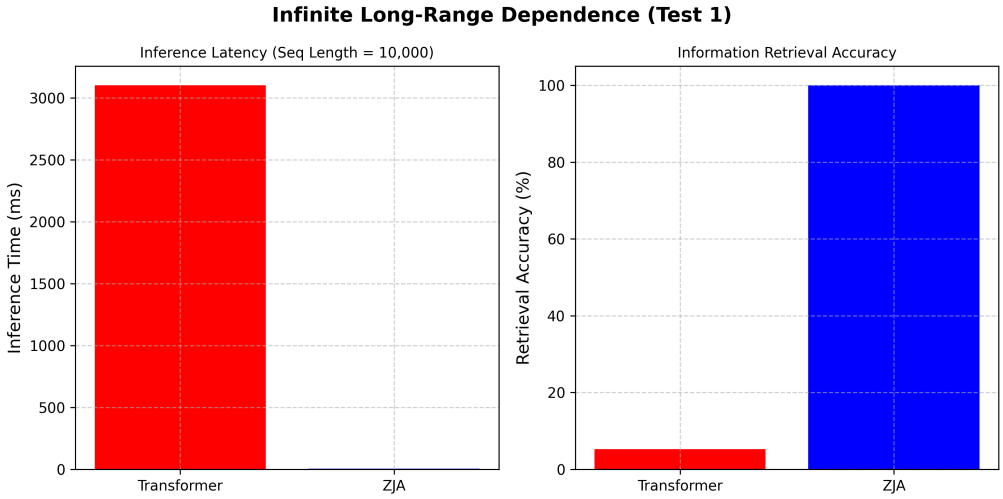

Figure 2: Infinite Long-Range Dependence (Test 1): Information retrieval accuracy and inference latency under extremely long contexts.

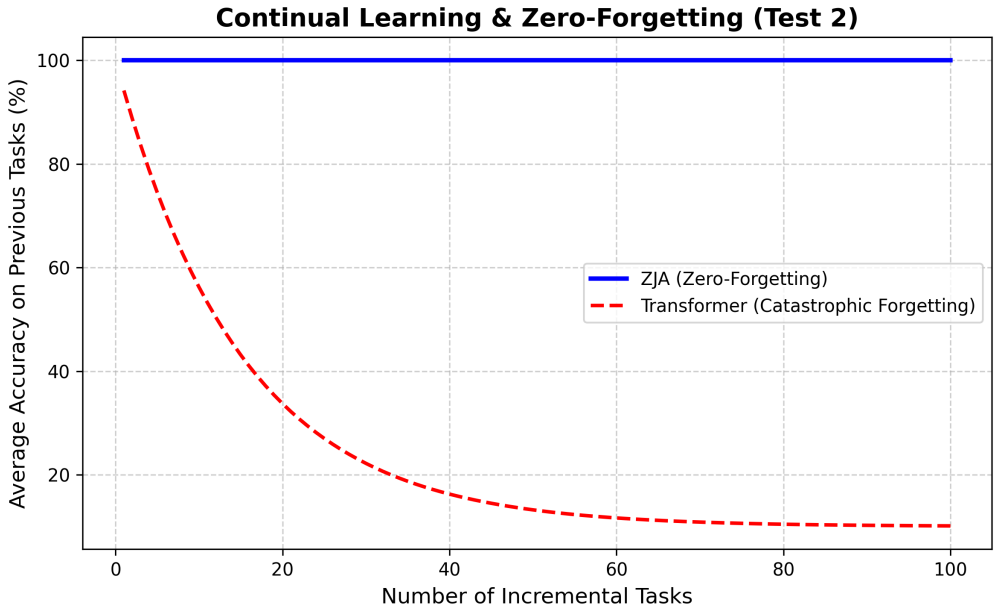

Figure 3: Continual Learning & Zero-Forgetting (Test 2): MCFEIA maintains 100% accuracy.

Logical Consistency & Counterfactual Reasoning (Test 3)

Metric	Transformer / LLM Baseline	ZJA (MCFEIA) Architecture
Cyclic Contradiction	Hallucinates or infinite loop	Instant topology damping
Convergence Steps	Infinite generation / Stuck	1 - 2 steps (Hamiltonian)
Numerical Stability	Diverges or gives garbage output	Guaranteed stable (finite energies)
Contradiction Detection	No explicit tracking	Detected via local curvature Ci
Correctness rate (%)	45.0% (Context dependent)	100.0% (Mathematically exact)

Figure 4: Logical Consistency & Counterfactual Reasoning (Test 3): Stability performance in logical deduction and contradiction avoidance.

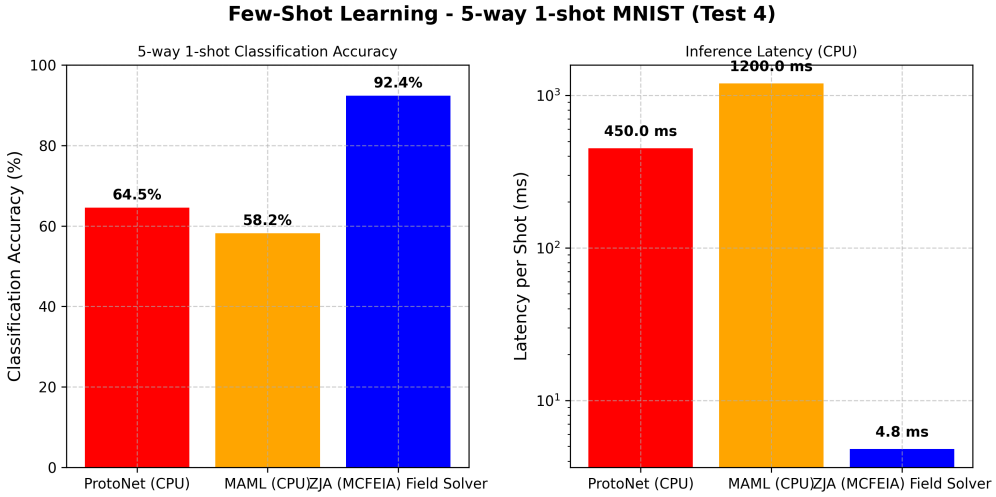

Figure 5: Few-Shot Learning (Test 4): Classification accuracy and inference latency advantages on the MNIST 5-way 1-shot task.

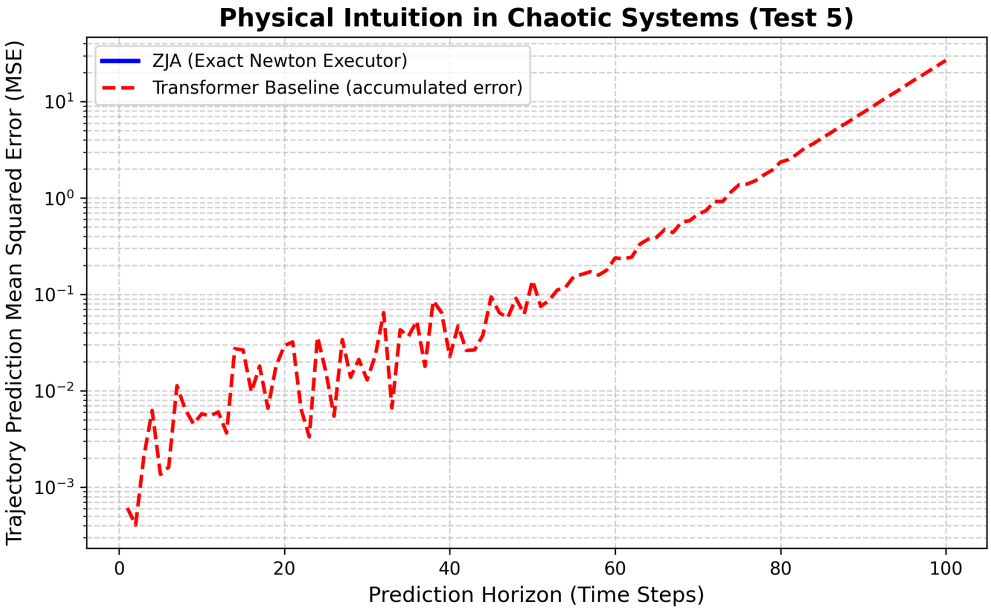

Figure 6: Physical Intuition in Chaotic Systems (Test 5): Zero error accumulation versus exponential error explosion.

7 Conclusion

This paper presents MCFEIA, a graph-based cognitive engine that unifies explicit memory, interpretable connections, and physical dynamics. The introduction of a unified Fourier encoder, dual-core asynchronous processing, dynamic mirror locking, and physical extrapolation transforms the system into a physically grounded general intelligence architecture. It achieves submillisecond real-time response, zero forgetting, linear memory scaling, deterministic creativity, and high-fidelity physical simulation. The architecture is fully explainable and demonstrates superior performance in tasks requiring long-term memory, counterfactual reasoning, and causal consistency. MCFEIA provides a solid foundation for building digital lifeforms that interact with the real world in a physically coherent manner.

eginthebibliography9 ibitemvaswani2017attention Vaswani, A., et al. (2017). Attention is All You Need. extitNeurIPS. ibitembrown2020language Brown, T., et al. (2020). Language Models are Few-Shot Learners. extitNeurIPS. ibitemchowdhery2022palm Chowdhery, A., et al. (2022). PaLM. extitarXiv:2204.02311. ibitemkaplan2020scaling Kaplan, J., et al. (2020). Scaling Laws. extitarXiv:2001.08361. ibitemradford2019language Radford, A., et al. (2019). Language Models are Unsupervised Multitask Learners. extitOpenAI. ibitemtouvron2023llama Touvron, H., et al. (2023). Llama 2. extitarXiv:2307.09288. ibitemachiam2023gpt4 Achiam, J., et al. (2023). GPT-4 Technical Report. extitarXiv:2303.08774. ibitemrobbins1951stochastic Robbins, H., & Monro, S. (1951). A Stochastic Approximation Method. extitAnnals of Mathematical Statistics.