perplexity

Encoding Tensor Networks and Neural Operations into Cellular Automata and Lenia Through Diffusion Dynamics

Foundational Synthesis of Tensor Networks and Cellular Automata

Tensor networks and cellular automata (CA) share a fundamental computational motif: the representation of complex global phenomena through structured local interactions. Tensor networks decompose high-dimensional data into contracted products of lower-dimensional tensors, enabling efficient representation of quantum states and machine learning models[1] [2]. Cellular automata achieve similar efficiency through localized update rules that propagate information across a lattice[3] [4]. The encoding challenge lies in mapping tensor contractions and neural operations onto CA/Lenia update rules while maintaining computational equivalence.

Diffusion models provide a natural framework for this encoding through their iterative denoising process, which mirrors the incremental state transitions of CA. By conceptualizing tensor operations as diffusion trajectories, we can train CA to approximate neural network computations:

1. **Tensor Contractions as Neighborhood Interactions**: Each tensor contraction
$$A_{ijk} = B_{il}C_{ljk}$$
can be modeled as a CA rule where cells at position
$$(i, j, k)$$
aggregate information from neighboring cells along dimension
$$l$$
[1:1] [5].

2. **Nonlinear Activations via Growth Functions**: Lenia's continuous growth functions
$$G$$
map aggregated neighborhood states (analogous to tensor products) to nonlinear updates, replicating activation functions like ReLU or sigmoid[4:1] [2:1].

Case Study: Matrix Multiplication in Lenia

Consider multiplying matrices
$$X$$
and
$$Y$$
using a Lenia-like system:

- **Kernel Design**: Convolution kernels
$$K_X$$
,

$$K_Y$$

encode row/column vectors.
- **State Update**:
$$a_{\mathbf{x}}^{t+1} = G\left(\sum_{\mathbf{y}} K_X(\mathbf{y}) \cdot X_{\mathbf{x}+\mathbf{y}}^t + \sum_{\mathbf{z}} K_Y(\mathbf{z}) \cdot Y_{\mathbf{x}+\mathbf{z}}^t \right)$$where

$$G$$

applies element-wise multiplication and summation[1:2] [4:2].

Diffusion-Driven Training of Neural Cellular Automata

Neural Cellular Automata (NCA) trained via diffusion processes bridge rule discovery and computational task performance[3:1] [6] [2:2]:

Diffusion as Rule Exploration

1. **Forward Process**: Inject Gaussian noise into CA rules

$$\mathcal{R}$$

, corrupting their functional form:

$$q(\mathcal{R}_t|\mathcal{R}_{t-1}) = \mathcal{N}(\mathcal{R}_t; \sqrt{1-\beta_t}\mathcal{R}_{t-1}, \beta_t\mathbf{I})$$

2. **Reverse Process**: Train a denoising network to recover task-optimal rules

$$\mathcal{R}_0$$

from

$$\mathcal{R}_T$$

, akin to Diff-NCA[6:1] [7].

This approach enables NCAs to:

- **Generalize Beyond Training Data**: Learn rules that extrapolate matrix operations to unseen dimensions[3:2] [1:3].
- **Preserve Symmetries**: Enforce rotational/translational invariance in tensor operations via equivariant kernels[3:3] [4:3].

Self-Reproducing Neural Networks as Cellular Automata

Von Neumann's self-replicating automata[8] and modern SeRANN models[9] [10] demonstrate how computational systems can encode their own replication. Integrating diffusion enhances this capability:

Architecture Encoding

1. **Genotype-Phenotype Mapping**:
   - **Genotype**: Bit-string encoding neural network weights/architecture.
   - **Phenotype**: CA rules that reconstruct the network via local interactions[9:1] [11].
2. **Diffusion-Guided Replication**:
   - Perturb genotypes with structured noise during replication.

- Denoising steps optimize for functional equivalence between parent/offspring networks[6:2] [10:1].

Example: Self-Replicating Perceptron

A minimal CA-based perceptron replicates via:

1. **State Propagation**: Input weights $\mathbf{w}$ are distributed across neighboring cells.
   2. **Rule Application**:
   $$w_i^{t+1} = ReLU\left(\sum_{j \in \mathcal{N}(i)} K_{rep}(j-i) w_j^t \right)$$where kernel K_{rep} ensures weight conservation[9:2] [11:1].

Lenia as a Continuous Tensor Network

Lenia's continuum framework naturally embeds tensor operations through:

Kernel-Tensor Equivalence

Radial convolution kernels $K(r)$ approximate tensor product expansions:

$$K(r) = \sum_{n=0}^{N} c_n e^{-(r-\mu_n)^2/\sigma_n^2}$$

where coefficients c_n parameterize arbitrary linear transformations[4:4] [2:3].

Dynamic Weight Modulation

Mirrored tensor surfaces[12] enable real-time adaptation:

- **CA-Driven Perturbation**: Auxiliary automata modulate Lenia's growth function parameters Δt , G to optimize task performance[12:1] [4:5].

Challenges and Future Directions

1. **Scalability**: Current CA models struggle with high-rank tensors due to neighborhood size limitations. Hybrid architectures combining local CA rules with global attention (ViTCA[2:4]) may mitigate this.

2. **Stochasticity vs. Precision**: Diffusion introduces noise beneficial for exploration but detrimental to exact tensor operations. Quasi-deterministic reverse processes could balance these[6:3] [7:1].
3. **Photonic Implementations**: PNCA[4:6] demonstrates optical realizations of CA-based networks, suggesting pathways for hardware-accelerated tensor-CA systems.

Conclusion

The synthesis of tensor networks, neural operations, and cellular automata through diffusion frameworks offers a revolutionary paradigm for adaptive, self-replicating computational systems. By encoding neural architectures into local update rules and leveraging diffusion for both training and replication, we achieve systems that balance biological inspiration with mathematical rigor. Future work may see CA-based chips executing diffusion-optimized tensor contractions, blurring the lines between classical computation and artificial life.

⁂

1. https://arxiv.org/pdf/2209.09098.pdf
2. https://openreview.net/forum?id=9t24EBSIZOa
3. https://pmc.ncbi.nlm.nih.gov/articles/PMC11078362/
4. https://www.nature.com/articles/s41377-024-01651-7
5. https://content.wolfram.com/sites/13/2018/04/26-4-2.pdf
6. https://arxiv.org/html/2401.06291v2
7. https://arxiv.org/html/2402.03961v1
8. https://fab.cba.mit.edu/classes/865.18/replication/Burks.pdf
9. https://journals.plos.org/ploscompbiol/article?id=10.1371%2Fjournal.pcbi.1012004
10. https://pmc.ncbi.nlm.nih.gov/articles/PMC11003675/
11. https://arxiv.org/html/2402.03961v2
12. https://www.reddit.com/r/MachineLearning/comments/1g0rhsx/r_cellular_automatondriven_mirrored_tensor/