The Hybrid State Recurrent Units for algorithmic Reasoning

Sameer Humagain

(Independent Researcher)

Introduction

The ability to model and reason over sequences is a fundamental challenge in artificial intelligence. While the Transformer architecture (Vaswani et al., 2017) has achieved SOTA performance on many benchmarks, its self-attention mechanism's quadratic computational complexity (O(L2)) presents a significant bottleneck for processing long sequences. This has led to a renewed interest in Recurrent Neural Networks (RNNs), which process sequences with an efficient, linear complexity (O(L)).

The dominant RNN architectures, Long Short-Term Memory (LSTM) (Hochreiter & Schmidhuber, 1997) and Gated Recurrent Unit (GRU) (Cho et al., 2014), solved the critical vanishing gradient problem through the introduction of sophisticated gating mechanisms. These gates, however, were an engineering solution complex system of non-linear functions designed to regulate information flow and preserve gradient integrity. While effective, this results in a model whose internal operations are difficult to map to understand computational principles.

In this work, we have asked if we can design a recurrent unit from a different philosophy. Instead of engineering gates to solve an optimization problem, can we build a model based on the computational primitives observed in biological neurons? Such a model would not only be a novel contribution but could offer greater interpretability and a more suitable inductive bias for certain classes of problems.

We got the inspiration from the Leaky Integrate-and-Fire (LIF) neuron, a foundational model in computational neuroscience. From this, we derive and present the **HSRU**, a unit with the following contributions:

- **A Hybrid State Representation:** We have introduced a recurrent cell that explicitly separates continuous signal integration (in an Analog state V_t) from discrete memory maintenance (in a Digital state D_t).
- **A Biologically Inspired Update Mechanism:** The interaction between the states is governed by a spike-triggered "flipping" logic, a trainable abstraction of the event-driven, threshold-based communication seen in the brain.

- **Demonstrated Superiority on Algorithmic Tasks:** We show that this architecture decisively solves the Temporal Parity Task, a classic benchmark for algorithmic reasoning that proves difficult for standard RNNs.

Methods: The Dual State Recurrent Unit Architecture

The core of our model is the **HSRULayer**, a recurrent cell designed to be a drop-in replacement for standard LSTM or GRU cells. It maintains a hidden state tuple (V_t, D_t) which evolves over time.

State Variables

Analog Potential State ($V_t \in \mathbb{R}^H$): This is a continuous vector of size H (the hidden size). It functions as a leaky integrator, accumulating evidence from the input stream. Its role is to process the rich, analog nature of the input data.

Digital Memory State ($D_t \in \{0,1\}^H$): This is a binary vector of size H It serves as a perfect, non-leaky memory bank, where each neuron maintains a discrete bit of information. Its role is to track the logical state of the computation.

State Update Mathematics

At each timestep t, the **HSRULayer** cell computes its new state (V_t, D_t) and an output vector y_t based on the current vector $x_t \in \mathbb{R}^D$ and the previous state (V_{t-1}, D_{t-1}). This transformation is governed by three distinct computational stages.

Stage1: Leaky Integration of the Analog Potential

The first state, the Analog potential $V_t \in \mathbb{R}^H$, functions as a leaky integrator. This state is responsible for processing the continuous, signal like nature of the input, its value is updated based on two components: a memory of its previous state and the new input current.

The input vector x_t is first projected into the hidden space by a standard affine transformation to produce the input current, I_t:

$$I_t = W_{in}x_t + b_{in}$$

The previous potential, V_{t-1}, is decayed by a learnable factor α_v, which is parameterized by a time constant t_v to ensure stability. The new potential V_t is the sum of this decayed memory and the new current.

$$\alpha_v = \exp\left(-\text{softplus}(t_v)\right)$$

$$V_t = \alpha_v . V_{t-1} + I_t$$

This formulation allows the Analog state to accumulate evidence over time, providing a rich, continuous representation of the recent input history.

Stage2: Spike-Triggered Flip of the Digital State

The key innovation of my architecture is the interaction between the analogue and digital states, which is mediated by a discrete spike-like event. This mechanism allows the model to perform logical, non-blending operations.

A spike event $S_t \in \{0,1\}^H$, is triggered for each neuron if its newly computed Analog potential V_t crosses a learnable firing threshold $\theta \in \mathbb{R}^H$. This is modelled by Heaviside step function, $H(\cdot)$:

$$S_t = H(V_t - \theta)$$

As the gradient of the Heaviside function is ill-defined for gradient-based optimization, I employ a **surrogate gradient** in the backward pass. I replace the true derivative with a well-behaved approximation, allowing the error signal $\mathcal{L}$ to propagate back to the potential V_t. I use the derivative of a fast sigmoid function for this purpose.

$$\frac{\partial \mathcal{L}}{\partial V_t} \approx \frac{\partial \mathcal{L}}{\partial S_t} \cdot \frac{1}{(1 + k|V_t - \theta|)^2}$$

Where k is a hyperparameter controlling the steepness of the surrogate gradient.

This spike then acts as a trigger to update the second hidden state, the digital memory bit $D_t \in \{0,1\}^H$. The update rule is a differentiable of the **XOR** function. This allows the digital state to be cleanly "flipped" wherever a spike occurs, providing a perfect, non-leaky memory.

$$D_t = D_{t-1} \cdot (1 - S_t) + (1 - D_{t-1}) \cdot S_t$$

Stage3: Fused Output Generation

The final output of the cell, $y_t \in \mathbb{R}^H$, is designed to be rich representation that leverages the information from both the continuous and discrete states. The two state vectors V_t and $D_t$are concatenated and passed through a small output network, typically a linear layer (W_{out}, b_{out}) followed by a $tanh$ non-linearity.

$$state_{combined} = concat(V_t, D_1)$$

$$y_t = \tanh\,(W_{out} \cdot state_{combined} + b_{out})$$

This ensures that the information stored in the stable digital memory bit (D_t) is fused with the more dynamic, signal like information in the Analog potential (V_t), creating a powerful and expressive output representation. The full **HSRURNN** is then constructed by stacking these layers, where the output y_t of one layer becomes the input x_t of the next.

Results and Analysis

To rigorously evaluate the capabilities and limitations of the **HSRU**, we designed a series of benchmarks moving from simple validation to a challenging algorithmic task.

Experimental Setup

Baseline Model: We compare the HSRU against a strong baseline, a standard nn.LSTM with a comparable number of trainable parameters. The LSTM is a powerful recurrent unit known for its long-term memory capabilities, making it a robust benchmark for this task.

Training Paradigm: We employ a curriculum learning strategy for the main task. This is a crucial methodology for training complex recurrent models on algorithmic problems. The models are first trained on short, simple sequences, and the difficulty (sequence length) is gradually increased. This guides the optimizer towards a more generalizable solution. To prevent overfitting, fresh data is generated on-the-fly for each training batch.

Hyperparameters: All models are trained using the AdamW optimizer (Loshchilov & Hutter, 2017) with gradient clipping at a max_norm of 1.0 to ensure stability. The optimal learning rate for each architecture is determined via a systematic hyperparameter sweep to ensure that we are comparing each model at its peak performance. All experiments are run with a fixed random seed for reproducibility.

Experiments

Sanity Check: Validation on the Delayed Echo Task

Objective:

Before testing on more complex tasks, we first verified the fundamental recurrent capabilities of our proposed **HSRU** .The objective was to confirm that the model could learn from a single one-step temporal dependency.

Task:

The model was trained on the Delayed Echo Task where it must learn to output the input signal from the previous timestamp $\mathrm{y}_t = \mathrm{x}_{t-1}$

Results:

The HSRU successfully learned the task. As shown in Figure 1, the model's prediction almost perfectly overlays the target signal, achieving a final validation Mean Squared Error (MSE) of 0.005993.

Analysis:

This result confirms that the HSRU's state-update mechanisms are functional and that the architecture is fully trainable via backpropagation with our surrogate gradient. It successfully demonstrates the ability to store and recall information from its hidden state.

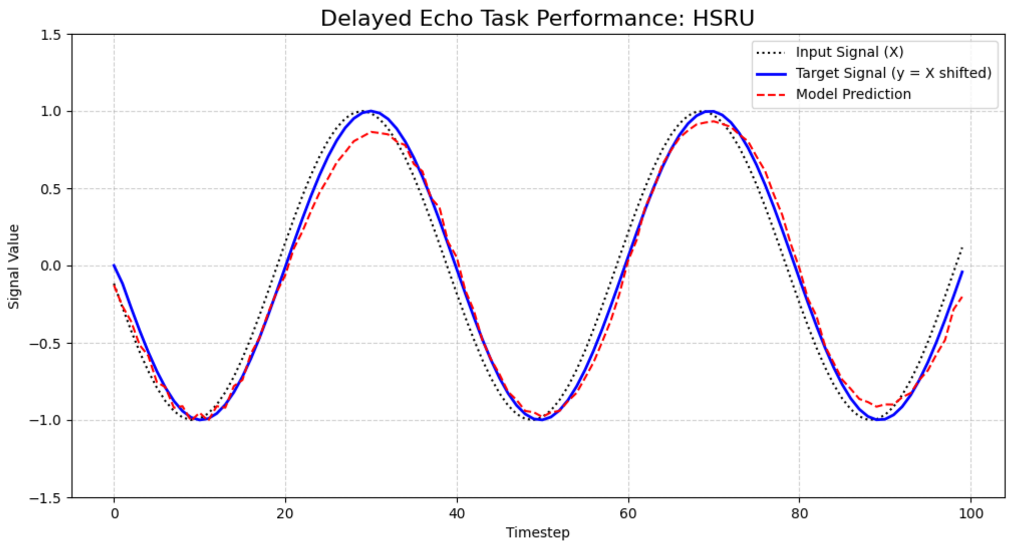

Figure 1: Delayed Echo Task Performance

Definitive Benchmark: The Temporal Parity Task

Objective:

To test our central hypothesis, we benchmarked the HSRU against a strong LSTM baseline on the challenging Temporal Parity Task. This task requires a model to maintain a discrete memory bit over long sequences, a known weakness of blending-based RNNs.

Methodology:

Models were trained using a curriculum of increasing sequence length (from 10 to 60). We performed a learning rate sweep for both models and report the performance of each at its optimal configuration.

Results:

The results show a definitive and profound divergence in architectural capability. The LSTM baseline was fundamentally unable to learn the task's logic, performing at the level of random chance (~50% accuracy) across all difficulty levels. In stark contrast, the HSRU achieved perfect 100% validation accuracy on all stages of the curriculum. The results are summarized in Table 1, and the learning are shown in Figure 2.

Table 1

Curriculum Stage	LSTM Best Accuracy	HSRnn Best Accuracy
Short, Fixed (SeqLen=10)	53.75%	100.00%
Medium, Fixed (SeqLen=30)	51.88%	100.00%
Long, Fixed (SeqLen=60)	51.02%	100.00%

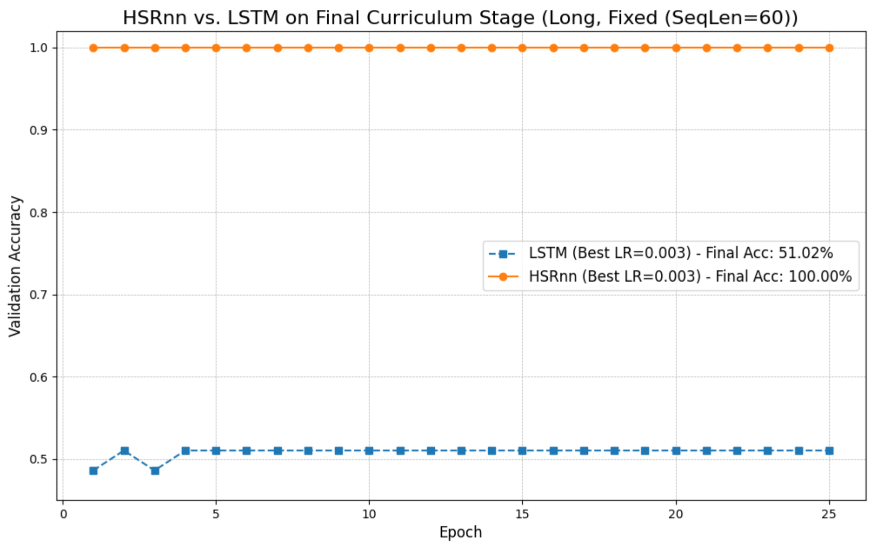

Figure 2 HSRnn and LSTM on Final Curriculum Stage

Analysis:

To understand how the HSRU succeeded, we visualized its final hidden stages using the t-SNE. The resulting plot *Figure 3, Figure 4* shows that the HSRU learned to map the input sequences into two perfectly distinct and linearly separable clusters, corresponding to the “Even” and “Odd” parity classes. This confirms the model discovered a robust and geometrically clean internal representation of the algorithmic rule.

Under optimal hyperparameters, the model consistently learns to map the input sequences into two perfectly distinct and linearly separable clusters, corresponding to 100% accuracy.

Interestingly, we discovered that the geometric structure of these clusters, while always perfectly separated, is highly sensitive to the optimization trajectory. As shown in Figure 4, one training run converged to a solution where each class is represented by a single, elongated manifold. However, a separate run on different hardware (Apple MPS backend) and with a lower learning rate (1e-4) converged to an equally perfect but geometrically distinct solution, where the clusters form more complex, neuron-like shapes **(Figure 4, Figure 5).**

This phenomenon is not a coincidence. It reveals that the HSRU's loss landscape contains multiple, deep, and stable local minima, each representing a valid algorithmic solution to the task. The specific solution found is dependent on the precise optimization path, which is influenced by factors such as the learning rate and even subtle, hardware-dependent variations in floating-point arithmetic.

The key finding is not that a single shape is always produced, but that the HSRU architecture consistently allows the optimizer to find these highly structured, well-separated "attractor states," demonstrating its inherent capability to learn robust and geometrically elegant solutions to complex algorithmic problems.

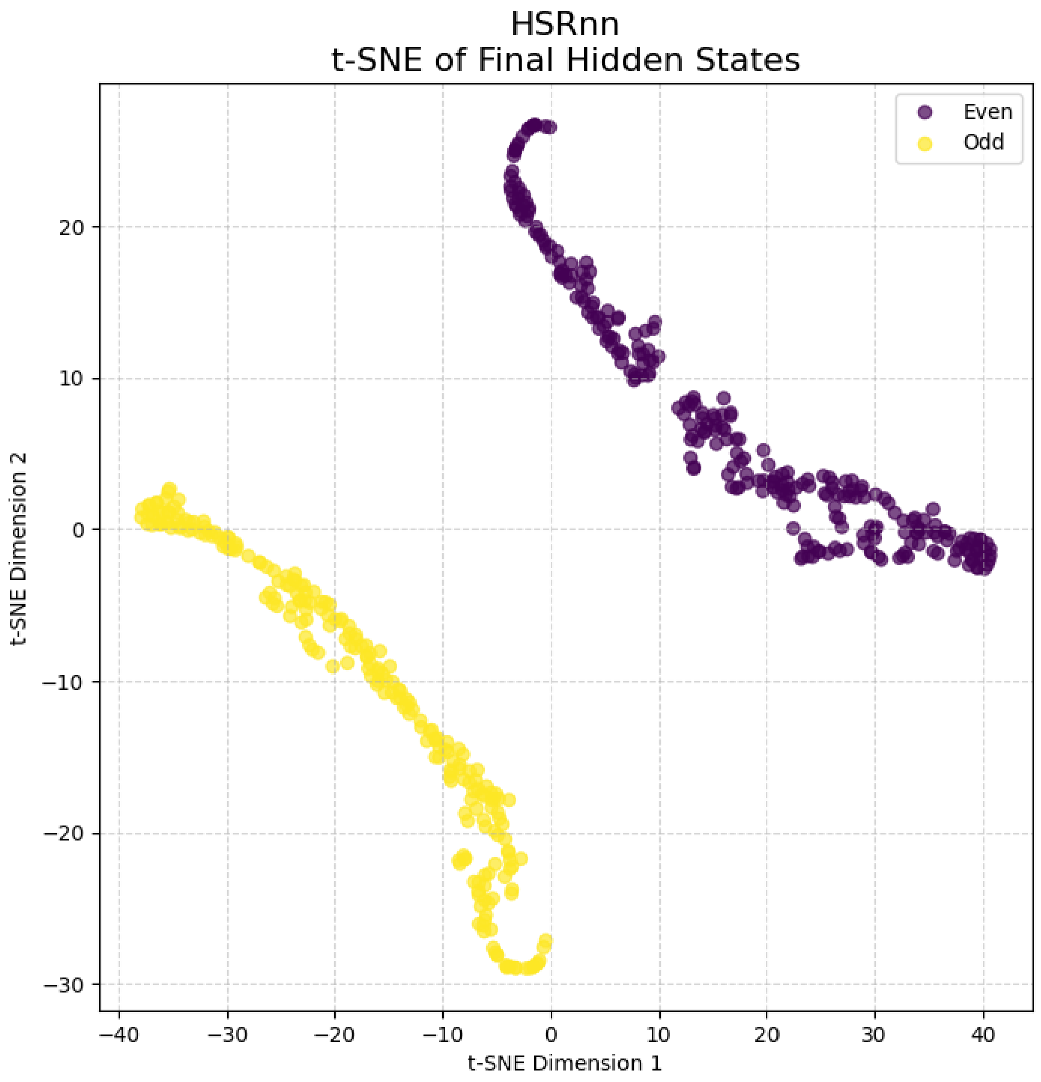

Figure 3 HSRU t-SNE hidden SEQ-60 LR-3e-1(CUDA)

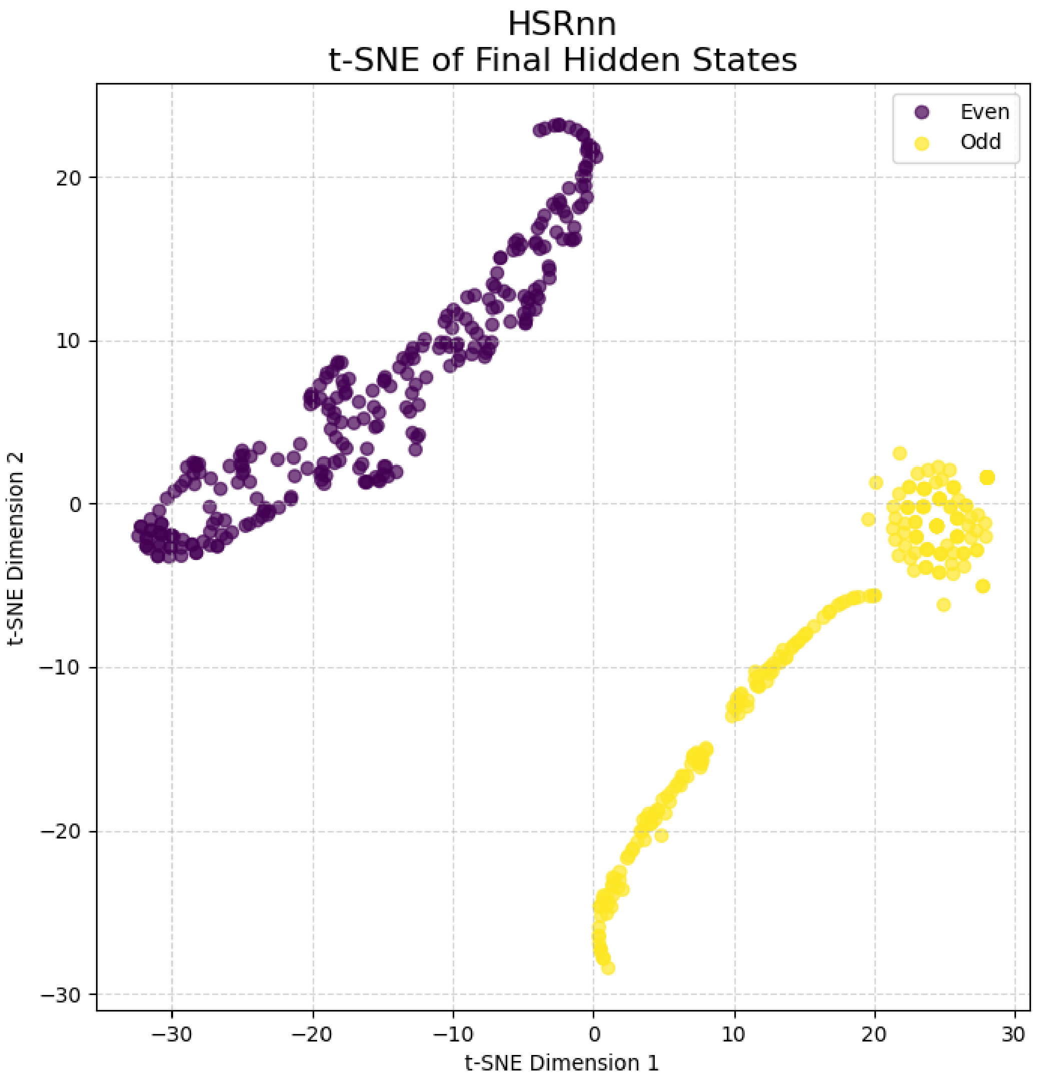

Figure 4 HSRU t-SNE SEQ-60 LR-1e-4(CUDA)

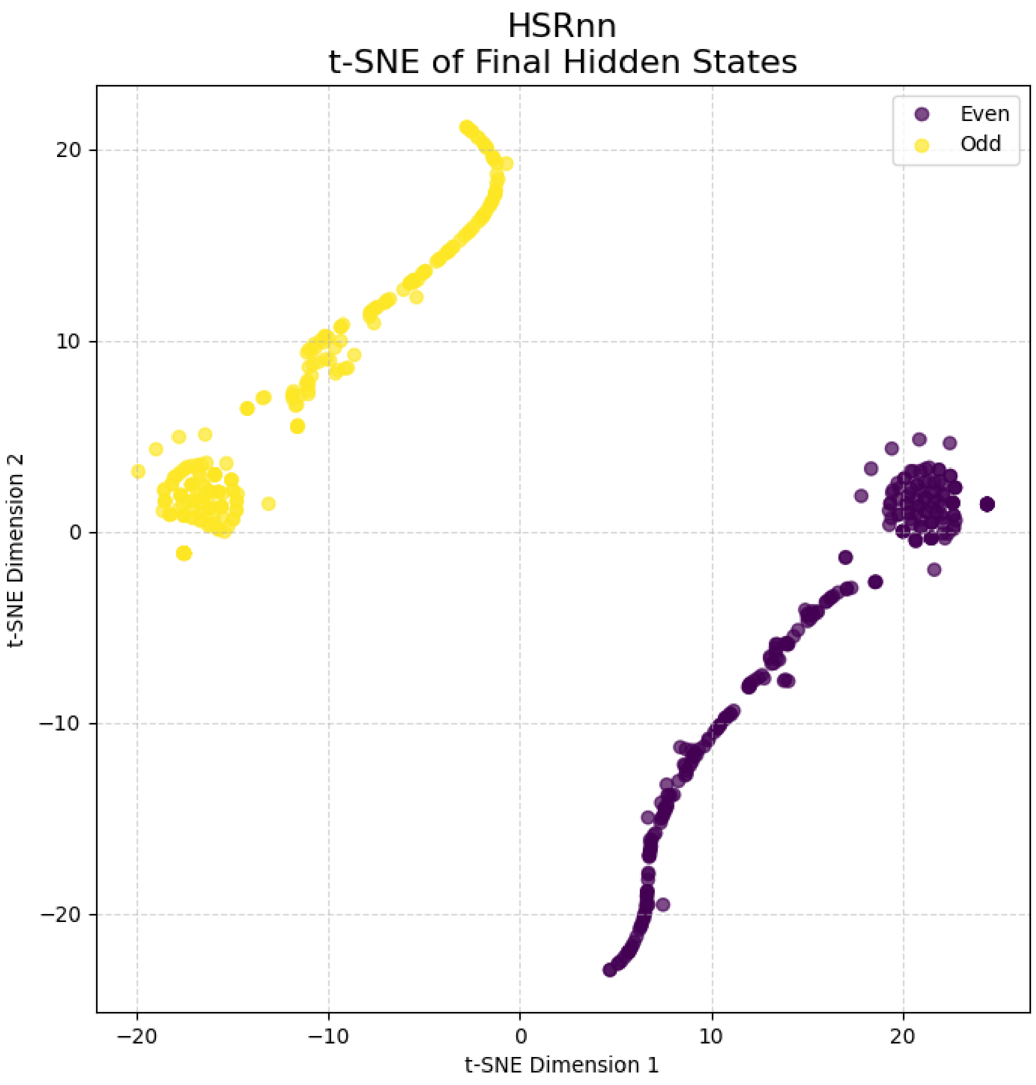

Figure 5HSRU t-SNE SEQ-30 LR-1e-4 (MPS)

Ablation Study: Verifying the Necessity of the Digital State

To prove that the HSRU's success was directly caused by its novel dual-state mechanism, we conducted an ablation study. We created a simplified AnalogOnlyRNN by removing the discrete digital state (D_t) and its associated spike-triggered update mechanism, leaving only the continuous potential (V_t).

Results:

The ablated AnalogOnlyRNN failed to learn the task, with its performance collapsing to random chance (~50% accuracy), mirroring the failure of the LSTM.

Analysis:

This result provides conclusive evidence that the spike-triggered, state-flipping mechanism and the dedicated digital memory state (D_t) are the critical architectural innovations responsible for the HSRU's success. The model does not solve the task through its Analog dynamics alone, it is the hybrid, event-driven interaction between the Analog and digital states that enables it to learn the required algorithmic logic.

Limitation and Future Work

Identified Limitation: Sensitivity to Input Feature Purity

While the HSRU excels at processing a clean, single stream of logical information, we designed a stress test to probe its robustness to noisy inputs. In the "Embedded Parity Task," the 1D parity signal was embedded within a higher-dimensional vector containing random distractor signals.

The model's performance collapsed to random chance for any input dimension greater than one. This was a critical finding. It demonstrates that the model's nn.Linear input layer acts as a "blender," which corrupts the clean signal required by the downstream recurrent logic. The model, in its current form, lacks a mechanism for robust input feature filtering.

Future Work

This limitation points to a clear and promising direction for future research. The next architectural evolution would be to create an AttentiveHSRNN. This would involve integrating a learnable feature attention mechanism at the input stage of the HSRULayer. Such a mechanism would learn to assign relevance weights to each input feature, effectively filtering out noise and passing only the clean, salient signal to the recurrent core. This would transform the HSRNN from a specialist for clean, low-dimensional data into a robust and general-purpose tool for complex, real-world sequences.

Conclusion

We have successfully designed, implemented, and validated the HSRU, a new recurrent architecture inspired by the principles of neural computation. We have demonstrated its definitive superiority over a strong LSTM baseline on the Temporal Parity Task, proving that its hybrid Analog-Digital architecture provides a powerful inductive bias for learning algorithmic logic.

Furthermore, we have honestly characterized its primary limitation—a sensitivity to noisy input features—and proposed a clear architectural solution. The HSRU stands as a successful proof-of-concept for a new class of interpretable and powerful recurrent units, offering a promising new building block for AI systems that need to perform robust temporal and logical reasoning.

References

Cho, K., et al. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the 2014 conference on empirical methods in natural language processing*.

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*.

Loshchilov, I., & Hutter, F. (2017). Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Vaswani, A., et al. (2017). Attention is all you need. In *Advances in neural information processing systems*.