Drama: Mamba-Enabled Model-Based Reinforcement Learning Is Sample and Parameter Efficient

April 16, 2025

Reference

Wenlong Wang, Ivana Dusparic, Yucheng Shi, Ke Zhang, and Vinny Cahill. Drama: Mamba-enabled model-based reinforcement learning is sample and parameter efficient, 2025. URL https://arxiv.org/abs/2410.08893.

Drama [Wang et al., 2025] Abstract

Challenges in Current World Models:

- RNNs: Struggle with vanishing gradients and long-term dependencies
- Transformers: Suffer from $O(n^2)$ memory and computational complexity

Their Approach - Drama:

- State space model (SSM)-based world model leveraging Mamba
- Achieves $O(n)$ memory and computational complexity
- Effectively captures long-term dependencies
- Novel sampling method to mitigate suboptimality from incorrect world models

Results:

- Competitive normalized score on *Atari100k* benchmark
- Only 7 million parameters in world model
- Trainable on standard laptops

Problem Formulation: POMDP

POMDP Environment:

- Agent observes image $\mathbf{O}_t \in \mathbb{O}$ instead of true state $s_t \in \mathbb{S}$
- Observation probability: $p(\mathbf{O}_t | s_t)$
- Discrete action space: $a_t \in \mathbb{A} = \{0, 1, \ldots, n\}$
- Transition dynamics: $p(s_{t+1}, r_t | s_t, a_t)$

Agent:

- Policy $\pi(\mathbf{O}_t; \boldsymbol{\theta}) : \mathbb{O} \to \mathbb{A}$
- Objective: Maximize $\mathbb{E} \sum_t \gamma^t r_t$

Model-Based Approach:

- Learn world model $f(\mathbf{O}_t, a_t; \omega)$ from experiences
- World model components: VAE, sequence model, reward/termination predictors
- "Imagination" process: Generate synthetic experiences for policy improvement

Drama Architecture

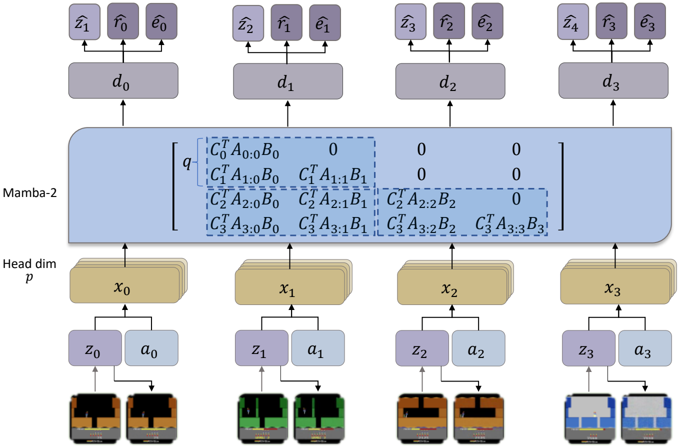

Figure: Drama architecture combining Mamba-based SSM world model with model-based RL

State Space Models (SSMs) - Basic Framework

Core Equations:

$$\boldsymbol{H}_t = \boldsymbol{A}\,\boldsymbol{H}_{t-1} + \boldsymbol{B}\,x_t$$
$$y_t = \boldsymbol{C}^\intercal\,\boldsymbol{H}_t$$

Key Components:

- $\boldsymbol{H}_t \in \mathbb{R}^n$: Hidden state summarizing input history
- x_t: Input at time t
- y_t: Output at time t
- $\boldsymbol{A}$: State transition matrix
- $\boldsymbol{B}$, $\boldsymbol{C}$: Input encoding and state decoding matrices

Advantages:

- Fixed-size hidden state regardless of sequence length
- $O(n)$ memory and computational complexity
- Strong theoretical foundation in control theory

Efficient SSM Structures

Diagonal Structure:

- Restrict $\boldsymbol{A}$ to be diagonal for computational efficiency
- Each dimension of hidden state updated independently

Time-Varying Matrices (Selective SSMs):

- Extend matrices to be time-dependent:

$$\tilde{\boldsymbol{A}} \in \mathbb{R}^{(T,N,N)}$$
$$\boldsymbol{B} \in \mathbb{R}^{(T,N)}$$
$$\boldsymbol{C} \in \mathbb{R}^{(T,N)}$$

- Allows model to adapt dynamics over sequence
- Enables capturing richer temporal patterns

Structured State Space Duality (SSD):

- Further constrains $\boldsymbol{A}$ as scalar multiple of identity matrix
- Simplifies model mathematically but limits expressiveness

Mamba-2: Enhanced Expressiveness

Multi-Head Technique:

- Treats each input channel as independent sequence
- Multiple "heads" learn different aspects of sequence dynamics
- Mitigates expressiveness limitations of simplified $\boldsymbol{A}$ matrix

Matrix Multiplication Reformulation:

$$y = \mathrm{SSM}(x;\ \tilde{\boldsymbol{A}}, \boldsymbol{B}, \boldsymbol{C}) = \mathbf{M}\, x$$

Transformation Matrix Entries:

$$\mathbf{M}_{j,i} = \begin{cases} \boldsymbol{C}_j^{\top}\, \boldsymbol{A}_{j:i}\, \boldsymbol{B}_i & \text{if } j \geq i, \\ 0 & \text{if } j < i, \end{cases}$$

where $\boldsymbol{A}_{j:i} = \boldsymbol{A}_j \boldsymbol{A}_{j-1} \ldots \boldsymbol{A}_{i+1}$

Benefits:

- 2-8× faster than original Mamba
- Highly GPU-efficient
- Maintains $O(n)$ scaling with sequence length

Connection to Causal Self-Attention

Semi-Separable Matrix Decomposition:

$$\mathbf{M} = \mathbf{L} \circ (\boldsymbol{C}\boldsymbol{B}^{\top})$$

Lower-Triangular Matrix Structure:

$$\mathbf{L} = \begin{bmatrix} 1 & & & & \\ a_1 & 1 & & & \\ a_2 a_1 & a_2 & 1 & & \\ \vdots & \vdots & \ddots & \ddots & \\ a_{T-1} \dots a_1 & a_{T-1} \dots a_2 & \dots & a_{T-1} & 1 \end{bmatrix}$$

where each $a_t \in [0, 1]$ is input-dependent

Relationship to Attention:

- $\mathbf{L}$ enforces causality similar to attention masks
- Replaces softmax normalization in traditional attention
- Creates direct link between SSMs and causal linear attention
- More efficient pathway to model dependencies

Summary of SSM Advantages

Control-Theoretic Foundation:

- Leverages established control theory principles
- Hidden state captures evolution of input history

Computational Efficiency:

- Linear $O(n)$ complexity in sequence length
- Diagonal and time-varying structures optimize computation
- Matrix multiplication formulation enables GPU acceleration

Modeling Capabilities:

- Effectively captures long-range dependencies
- Multi-head approach enhances expressiveness
- Connection to attention mechanisms without quadratic complexity

Practical Benefits:

- Enables processing of longer sequences
- Reduced memory requirements
- Faster training and inference

Latent Space Sequence Modeling with Mamba-2

Key Components:

- **Latent Variable $\boldsymbol{z}_t$**: Lower-dimensional encoding of observations
- **Deterministic State $\boldsymbol{d}_t$**: Distinct from hidden states used by SSMs to track dynamics
- **Action Integration**: Concatenates latent $\boldsymbol{z}_t$ with action a_t and projects through a fully connected layer

Sequence Construction:

$$\begin{array}{ll} \text{Sequence model:} & \boldsymbol{d}_t = f(\boldsymbol{z}_{t-l:t}, a_{t-l:t}; \omega) \\ \text{Latent variable predictor:} & \hat{\boldsymbol{z}}_{t+1} \sim p(\hat{\boldsymbol{z}}_{t+1} | \boldsymbol{d}_t; \omega) \end{array}$$

Latent Space Sequence Modeling with Mamba-2 (cont.)

Implementation Features:

- Processes batches $\mathbf{O} \in [0, 255]^{(b,l,h,w,c)}$ from experience buffer $\mathcal{E}$
- Encodes to latent representations $\mathbf{Z} \in \mathbb{R}^{(b,l,d)}$
- Eliminates sequential dependencies: $\boldsymbol{z}_{t+1}$ depends solely on observation $\mathbf{O}_{t+1}$
- Linear scaling with sequence length $O(l)$ rather than quadratic

Processing Pipeline:

- Input tensor $\mathbf{X}_{b,:l,d}$ processed into hidden states $\mathbf{H} \in \mathbb{R}^{(b,l-1,n)}$ with fixed dimension n
- Hidden states remapped to deterministic state sequence $\mathbf{D}_{b,:l,d}$ using time-varying parameters
- Multi-headed approach: latent dimension d split into $\frac{d}{p}$ independently processed heads

Mamba-2 Architecture for Sequence Modeling

Structured Transformation Matrix:

- Semiseparable lower triangular matrix ensures causality
- Decomposed into $q \times q$ specialized blocks for:
  - Short-range causal attention
  - Hidden state transformations
- Leverages matrix multiplication for efficient computation

Advantages:

- Efficiently captures temporal dynamics in latent space
- Hidden states operate in fixed dimension (unlike attention where state scales with sequence length)
- Combines benefits of state-space models with attention-like mechanisms
- Achieves linear computational complexity in sequence length l

Behaviour Policy Learning in Imagination

Imagination Process:

- Autoregressive simulation driven by Mamba sequence model
- Samples b_{img} trajectories (length l_{img}) from replay buffer
- Extends each trajectory with h additional simulated steps
- Automatic state reset at episode boundaries (no manual intervention)

Key Advantages:

- Decouples inference parameter updates from sequence length
- Significantly accelerates imagination process
- Rich state representation: $\hat{\boldsymbol{z}}_t$ (predictions) + $\boldsymbol{d}_t$ (history)
- Actor-critic architecture with specialized normalization techniques

Dynamic Frequency-Based Sampling (DFS)

Motivation:

- Early in training, world model has significant inaccuracies
- Unreliable predictions lead to reward underestimation and poor exploration
- Need to ensure behavior policy uses transitions the world model understands well

Dual Tracking System:

- World model tracking: $\boldsymbol{v} = (v_1, v_2, \ldots, v_{|\mathcal{E}|})$
  - v_i counts how often transition i sampled for world model training
- Behavior policy tracking: $\boldsymbol{b} = (b_1, b_2, \ldots, b_{|\mathcal{E}|})$
  - b_i counts how often transition i used for actor-critic learning

DFS Implementation Details

World Model Sampling Probabilities:

$$(p_1, p_2, \ldots, p_{|\mathcal{E}|}) = \texttt{softmax}(-\boldsymbol{v})$$

Behavior Policy Sampling Probabilities:

$$(q_1, q_2, \ldots, q_{|\mathcal{E}|}) = \texttt{softmax}(f(\boldsymbol{v}, \boldsymbol{b})),$$
$$\text{where } f(\boldsymbol{v}, \boldsymbol{b}) = \boldsymbol{v} - \boldsymbol{b} - \max(0, \boldsymbol{v} - \boldsymbol{b})$$

Piecewise Behavior:

- When $v_i \geq b_i$: $f(v_i, b_i) = 0$
  - World model sufficiently trained on this transition
  - Reliable for imagination process
- When $v_i < b_i$: $f(v_i, b_i) = v_i - b_i < 0$
  - Lower probability assigned via softmax
  - Avoids using transitions with unreliable world model predictions

Benefits:

- Ensures imagination based on reliable world model predictions
- Prevents overfitting to poorly learned experiences
- Dynamically adapts as training progresses
- Balances exploration and exploitation in model-based RL

Atari100k Benchmark Results

	Random	Human	PPO	SimPLe	SPR	TWM	IRIS	STROM	DreamerV3	DramaXS
Alien	228	7128	276	617	842	675	420	984	**1118**	820
Amidar	6	1720	26	74	180	122	143	**205**	97	131
Assault	222	742	327	527	566	683	**1524**	801	683	539
Asterix	210	8503	292	1128	962	1117	854	1028	1062	**1632**
BankHeist	14	753	14	34	345	467	53	**641**	398	137
BattleZone	2360	37188	2233	4031	14834	5068	13074	13540	**20300**	10860
Boxing	0	12	3	8	36	78	70	80	**82**	78
Breakout	2	30	3	16	20	20	**84**	16	10	7
ChopperCommand	811	7388	1005	979	946	1697	1565	1888	**2222**	1642
CrazyClimber	10780	35829	14675	62584	36700	71820	59324	66776	**86225**	83931
DemonAttack	152	1971	160	208	518	350	**2034**	165	577	201
Freeway	0	30	2	17	19	24	31	**34**	0	15
Frostbite	65	4335	127	237	1171	1476	259	1316	**3377**	785
Gopher	258	2412	368	597	661	1675	2236	**8240**	2160	2757
Hero	1027	30826	2596	2657	5859	7254	7037	11044	**13354**	7946
Jamesbond	29	303	41	100	366	362	463	509	**540**	372
Kangaroo	52	3035	55	51	3617	1240	838	**4208**	2643	1384
Krull	1598	2666	3222	2205	3682	6349	6616	8413	8171	**9693**
KungFuMaster	258	22736	2090	14862	14783	24555	21760	**26183**	25900	23920
MsPacman	307	6952	366	1480	1318	1588	999	**2673**	1521	2270
Pong	-21	15	-20	13	-5	**19**	15	11	-4	15
PrivateEye	25	69571	100	35	86	87	100	**7781**	3238	90
Qbert	164	13455	317	1289	866	3331	746	**4522**	2921	796
RoadRunner	12	7845	602	5641	12213	9109	9615	17564	**19230**	14020
Seaquest	68	42055	305	683	558	774	661	525	**962**	497
UpNDown	533	11693	1502	3350	10859	15982	3546	7985	**46910**	7387
Normalised Mean (%)	0	100	11	33	62	96	105	127	125	105
Normalised Median (%)	0	100	3	13	40	51	29	58	49	27

Ablation 1/3: DFS

DFS vs. Uniform Sampling:

- DFS achieves 105% normalized mean score (vs. uniform's 80%)
- Similar median performance (27% vs. 28%)
- DFS shows significant advantages in games with evolving dynamics:
  - `Alien`, `Asterix`, `BankHeist`, `Seaquest`
- Strong performance in opponent-based games:
  - `Boxing`, `Pong` (exploiting opponent AI weaknesses)
- Less effective in games with early-accessible critical dynamics:
  - `Breakout`, `KungFuMaster`

DFS vs. Uniform Sampling Performance

Game	Random	Human	DFS	Uniform
Alien	228	7128	**820**	696
Amidar	6	1720	131	**154**
Assault	222	742	**539**	511
Asterix	210	8503	**1632**	1045
BankHeist	14	753	**137**	52
BattleZone	2360	37188	10860	**10900**
Boxing	0	12	**78**	49
Breakout	2	30	7	**11**
ChopperCommand	811	7388	**1642**	1083
CrazyClimber	10780	35829	**83931**	77140
DemonAttack	152	1971	**201**	151
Freeway	0	30	**15**	15
Frostbite	65	4335	785	**975**
Gopher	258	2412	**2757**	2289
Hero	1027	30826	**7946**	7564
Jamesbond	29	303	**372**	363
Kangaroo	52	3035	**1384**	620
Krull	1598	2666	**9693**	7553
KungFuMaster	258	22736	23920	**24030**
MsPacman	307	6952	2270	**2508**
Pong	-21	15	**15**	3
PrivateEye	25	69571	**90**	76
Qbert	164	13455	796	**939**
RoadRunner	12	7845	**14020**	9328
Seaquest	68	42055	**497**	384
UpNDown	533	11693	**7387**	5756
Normalised Mean (%)	0	100	105	80
Normalised Median (%)	0	100	27	28

Ablation 2/3: Mamba vs. Mamba-2

Mamba-2 restricts the diagonal matrix $\boldsymbol{A}$ for efficiency. We compare Mamba-2 and Mamba as world model backbones using identical hyperparameters with DFS.

The following figure shows Mamba-2 outperforming Mamba in `Krull`, `Boxing` and `Freeway`. In `Krull`, Mamba plateaus when failing to rescue the princess, while Mamba-2 succeeds. In sparse-reward `Freeway`, only the DFS+Mamba-2 combination achieves positive results.

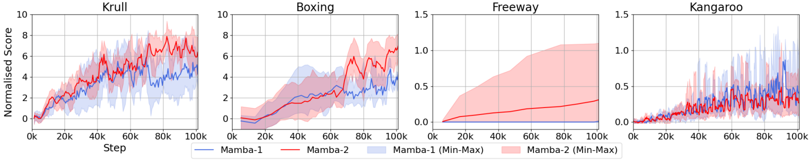

Figure: Mamba vs. Mamba-2. Mamba2 has shown a superior performance to Mamba in three out of four games. Both Mamba and Mamba-2 use DFS in this experiment.

Ablation 3/3: Sequence models for long-sequence predictability tasks

Environment:

- 5×5 grid with outer walls and 3×3 traversable area
- Red agent moves based on actions, yellow fixed goal
- Positions re-randomized when agent reaches goal

Sequence Construction:

- Each frame: $l_f = 5^2 + 1 = 26$ tokens (grid cells + action)
- Short sequence: $l = 8 \times l_f = 208$ tokens
- Long sequence: $l = 64 \times l_f = 1664$ tokens

Learning Objectives:

- Reconstruct grid layout
- Track agent position based on movement history
- Capture long-term dependencies

Models Compared:

- Mamba-2 & Mamba
- GRU
- Transformer

Grid World Experiment: Results

Method	l	Training Time (ms)	Memory Usage (%)	Error (%)
Mamba-2	208	25	13	15.6 ± 2.6
	1664	214	55	14.2 ± 0.3
Mamba	208	34	14	13.9 ± 0.4
	1664	299	52	14.0 ± 0.4
GRU	208	75	66	21.3 ± 0.3
	1664	628	68	34.7 ± 25.4
Transformer	208	45	17	75.0 ± 1.1
	1664	-	OOM	-

Table: Performance comparison of different methods in the grid world environment. Memory usage is reported as a percentage of an 8GB GPU. The error is represented as the mean $\pm$ standard deviation. The training time refers to the average duration per training step. Notably, the Transformer encounters an out-of-memory (OOM) error during training with long sequences. All experiments are conducted on a laptop.

Key Findings:

- Mamba-2 shows best overall performance with lowest training time
- Both Mamba variants maintain low reconstruction error on long sequences
- GRU shows increased error and training time with longer sequences
- Transformer runs out of memory (OOM) on long sequences
- Results confirm Mamba-based models' strong capability for long-sequence modeling in MBRL