Transformers are Sample-Efficient World Models

March 5, 2025

Reference

Vincent Micheli, Eloi Alonso, and François Fleuret. Transformers are sample-efficient world models, 2023. URL https://arxiv.org/abs/2209.00588.

Sample Efficiency Challenge in World Models

- Common drawback: Extremely low sample efficiency
- Experience requirements:
  - DreamerV2: Months of gameplay for Atari 2600
  - OpenAI Five: Thousands of years for Dota2
- Real-world limitations:
  - Cannot always speed up training environments
  - Cost and safety considerations
- **Key insight:** Sample efficiency is crucial for practical deployment of deep RL agents

World Models: Progress and Challenges

- Model-based methods offer promising path to data efficiency
- Recent applications of world models:
  - Pure representation learning
  - Lookahead search
  - Learning in imagination
- Learning in imagination advantages:
  - Frees agent from sample efficiency constraints
  - But requires highly accurate world models
- Key milestones:
  - Early success in toy environments
  - SimPLe: Progress on Atari 100k benchmark
  - DreamerV2: Current best but requires 200M frames
- **Challenge:** Need better architectures for complex environments with limited samples

Transformers: A Promising Direction for World Models

- Transformers have shown remarkable success across domains:
  - Natural Language Processing
  - Computer Vision
  - Offline Reinforcement Learning
- Key advantages:
  - Effective with high-dimensional data
  - Self-supervised learning capabilities
  - Strong performance with discrete tokens
- Addressing computational challenges:
  - Direct pixel-to-token conversion impractical
  - Solution: Discrete autoencoders (VQGAN, DALL-E)
  - Maps raw pixels to manageable token sets
- **Opportunity:** Transformer-based architectures show promise for efficient world modeling

IRIS: Transformer-Based World Model for Sample-Efficient RL

- IRIS (Imagination with auto-Regression over an Inner Speech) Micheli et al. [2023]:
  - Combines discrete autoencoder with autoregressive Transformer
  - Learns through simulated trajectories in imagination
  - Treats dynamics learning as sequence modeling problem
- Key innovations:
  - Autoencoder creates "language" of image tokens
  - Transformer composes tokens temporally
  - Minimal hyperparameter tuning needed
- Impressive results on Atari 100k benchmark:
  - 1.046 mean human-normalized score
  - Superhuman on 10/26 games
  - Only 2 hours of real experience needed
  - Outperforms recent sample-efficient methods

Learning in Imagination: Three Components

- Our approach follows three standard components for imagination-based learning:
  - `collect_experience` from environment
  - `update_world_model` from collected data
  - `update_behavior` within learned model
- Key characteristics:
  - Agent learns to act exclusively in world model
  - Real experience only used for dynamics learning
  - Follows successful approaches like Dreamer V1/V2

Mathematical Breakdown: Encoding

Given an input image:

$$x_t \in \mathbb{R}^{h \times w \times 3},$$

The encoder E (CNN) maps it to continuous latent vectors:

$$y_t = \{y_t^k\}_{k=1}^{K} \subset \mathbb{R}^d,$$

Formally:

$$E(x_t) = y_t \in \mathbb{R}^{K \times d}.$$

Mathematical Breakdown: Quantization to Discrete Tokens

Define embedding table (codebook):

$$\mathcal{E} = \{e_i\}_{i=1}^{N} \subset \mathbb{R}^d.$$

For each latent vector y_t^k, select closest embedding vector:

$$z_t^k = \arg \min_{i \in \{1,\ldots,N\}} \|y_t^k - e_i\|_2,$$

Resulting discrete token sequence:

$$z_t = (z_t^1, z_t^2, \ldots, z_t^K) \in \{1, \ldots, N\}^K.$$

Mathematical Breakdown: Decoding

Decoder D maps token sequence back to image space:

$$\hat{x}_t = D(z_t),$$

Formally:

$$D : \{1, \ldots, N\}^K \to \mathbb{R}^{h \times w \times 3}.$$

Mathematical Breakdown: Loss Functions and Training

Let's break down the loss function into its four components:

1. **Reconstruction Loss**: Measures the pixel-level difference between input x and reconstruction $D(z)$.

$$\|x - D(z)\|_1$$

2. **Commitment Loss Terms**: Align encoder output $E(x)$ with embedding $\mathcal{E}(z)$. The stop-gradient operator $\mathrm{sg}(\cdot)$ decouples encoder and embedding updates, facilitating discrete latent space learning.

$$\|\mathrm{sg}(E(x)) - \mathcal{E}(z)\|_2^2 + \|\mathrm{sg}(\mathcal{E}(z)) - E(x)\|_2^2$$

3. **Perceptual Loss**: Compares high-level features using a pre-trained network (e.g., VGG), ensuring semantic and perceptual similarity.

$$\mathcal{L}_{\text{perceptual}}(x, D(z))$$

The total loss function is given by:

$$\mathcal{L}(E, D, \mathcal{E}) = \|x - D(z)\|_1 + \|\mathrm{sg}(E(x)) - \mathcal{E}(z)\|_2^2 + \|\mathrm{sg}(\mathcal{E}(z)) - E(x)\|_2^2 + \mathcal{L}_{\text{perceptual}}(x, D(z))$$

This combination encourages accurate reconstruction, consistent discrete embeddings, and high perceptual quality.

Mathematical Breakdown: Straight-Through Estimator

Quantization step is non-differentiable:

$$z_t^k = \arg\min_i \|y_t^k - e_i\|_2$$

Straight-through estimator enables gradient flow:

- Copies gradients from decoder D back to encoder E
- Treats quantization as identity during backward pass

Allows end-to-end training despite discrete quantization.

Transformer-Based Sequence Modeling

The Transformer G learns environment dynamics by modeling sequences of encoded observations and actions as a "language":

Sequence Representation:

$$(x_0, a_0, x_1, a_1, \ldots, x_t, a_t)$$

Each image x_t is encoded into discrete tokens:

$$(z_t^1, z_t^2, \ldots, z_t^K)$$

Resulting interleaved sequence:

$$(z_0^1, \ldots, z_0^K, a_0, z_1^1, \ldots, z_1^K, a_1, \ldots, z_t^1, \ldots, z_t^K, a_t)$$

Predicted Distributions

At each timestep, Transformer predicts three key elements:

- **Transition (next state tokens):**

$$p_G\left(\hat{z}_{t+1} \mid z_{\leq t}, a_{\leq t}\right)$$

- **Reward:**

$$p_G\left(\hat{r}_t \mid z_{\leq t}, a_{\leq t}\right)$$

- **Termination (episode end):**

$$p_G\left(\hat{d}_t \mid z_{\leq t}, a_{\leq t}\right)$$

Enables complete modeling of environment dynamics.

Autoregressive Prediction with Transformer G

This is how $p_G(\hat{z}_{t+1} \mid z_{\leq t}, a_{\leq t})$ works:

Transformer G operates at the token level, predicting future states autoregressively:

To predict next frame tokens $\hat{z}_{t+1}$:

$$p_G(\hat{z}^k_{t+1} \mid z_{\leq t}, a_{\leq t}, z^{<k}_{t+1})$$

Each individual token prediction conditioned on:

- All previous tokens and actions $(z_{\leq t}, a_{\leq t})$
- Previously predicted tokens of current frame $z^{<k}_{t+1}$

This token-level prediction allows the Transformer to learn fine-grained temporal dependencies and dynamics.

Training the Transformer

Transformer G trained via self-supervised learning on segments of length L:

- **Self-Supervised Learning:**
  - Predicts future parts of sequences from past experiences
  - No labeled data required
- **Loss Functions:**
  - **Transition:** Cross-entropy loss (autoregressive token prediction)

$$\mathcal{L}_{\text{trans}} = \sum_{t} \sum_{k=1}^{K} \text{CE}\Big(p_G(\hat{z}_{t+1}^{k} | z_{\leq t}, a_{\leq t}, z_{t+1}^{<k}), z_{t+1}^{k}\Big)$$

  - **Termination:** Cross-entropy loss (episode end prediction)

$$\mathcal{L}_{\text{term}} = \sum_{t} \text{CE}\Big(p_G(\hat{d}_t | z_{\leq t}, a_{\leq t}), d_t\Big)$$

  - **Reward:** MSE (continuous) or cross-entropy (discrete)

$$\mathcal{L}_{\text{reward}} = \begin{cases} \text{MSE}(p_G(\hat{r}_t | z_{\leq t}, a_{\leq t}), r_t), & \text{continuous} \\ \text{CE}(p_G(\hat{r}_t | z_{\leq t}, a_{\leq t}), r_t), & \text{discrete} \end{cases}$$

Minimizing these losses ensures accurate modeling of transitions, rewards, and termination.

Actor-Critic Learning Objectives

We follow Dreamer V2 for actor-critic training, using the generic λ-return to balance bias and variance:

Given imagined trajectory $(\hat{x}_0, a_0, \hat{r}_0, \hat{d}_0, \ldots, \hat{x}_H)$, define recursively:

$$\Lambda_t = \begin{cases} \hat{r}_t + \gamma(1-\hat{d}_t)\Big[(1-\lambda)V(\hat{x}_{t+1}) + \lambda\Lambda_{t+1}\Big], & t < H \\ V(\hat{x}_H), & t = H \end{cases}$$

Value network V minimizes squared difference with λ-returns:

$$\mathcal{L}_V = \mathbb{E}_\pi\Big[\sum_{t=0}^{H-1}\big(V(\hat{x}_t) - \mathrm{sg}(\Lambda_t)\big)^2\Big]$$

Actor policy π is trained to maximize expected returns with entropy regularization:

$$\mathcal{L}_\pi = -\mathbb{E}_\pi\Big[\sum_{t=0}^{H-1}\log(\pi(a_t|\hat{x}_{\le t}))\,\mathrm{sg}(\Lambda_t - V(\hat{x}_t)) + \eta\,\mathcal{H}(\pi(a_t|\hat{x}_{\le t}))\Big]$$

Note: Actor-critic objectives are essentially identical to Dreamer V2.

Atari 100k Benchmark: Overview

- **Benchmark Description:**
  - 26 Atari games with only 100k actions ($\approx$ 2 hours of gameplay)
  - 500 $\times$ fewer steps than standard Atari benchmarks (50M steps)
  - Tests sample efficiency in reinforcement learning
- **Compared Methods:**
  - **Without search:** SimPLe, CURL, DrQ, SPR
  - **With search:** MuZero, EfficientZero (use planning)

IRIS Performance Results

- **Key Achievements:**
  - **Superhuman performance:** 1.046 mean human-normalized score
  - Outperforms human players in 10/26 games
  - New state-of-the-art among methods without search
  - Even surpasses MuZero on this sample-efficient benchmark
  - (But didn't surpass EfficientZero)
- **Game-Specific Performance:**
  - Excels in games with predictable dynamics (Pong, Breakout, Boxing)
  - Struggles with games requiring rare event discovery (Frostbite, Krull)

Insights and Challenges

- **World Model Importance:**
  - Agent learns entirely in imagination
  - Accurate prediction of objects, rewards, and terminations is crucial
  - Can generate multiple plausible futures under uncertainty
- **Key Challenges:**
  - **"Double Exploration" Problem:**
    - Agent must discover rare events/mechanics
    - World model must learn them before policy can exploit
  - Complex visual details require higher model capacity
- **Conclusion:** World models enable sample-efficient RL when they accurately capture environment dynamics