PoE-World: Compositional World Modeling with Products of Programmatic Experts

Wasu Top Piriyakulkij[1], Yichao Liang[2], Hao Tang[1],
Adrian Weller[2,3], Marta Kryven[4], Kevin Ellis[1]

[1]Cornell University, [2]University of Cambridge, [3]The Alan Turing Institute, [4]Dalhousie University

Neurips 2025 (Spotlight)

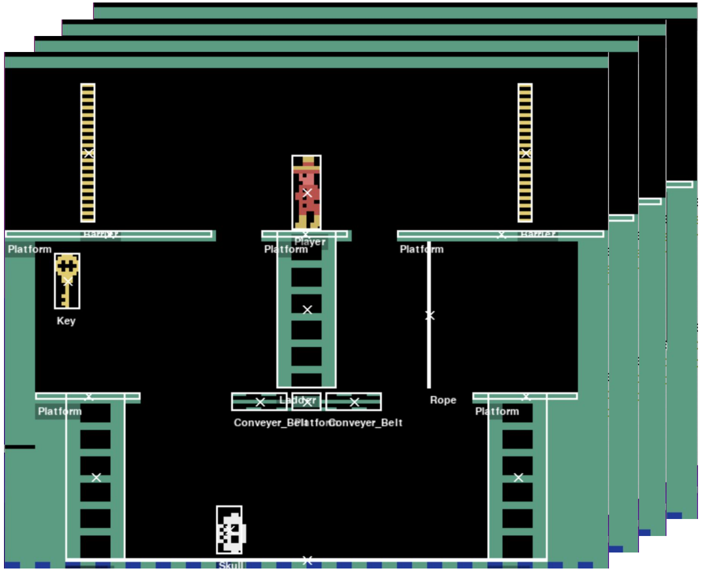
Check out the project website!

Link: https://topwasu.github.io/poe-world

- We made a one-minute video that shows our agent in-action!

World Modeling of a POMDP environment

Past observations $s_{1:t}$



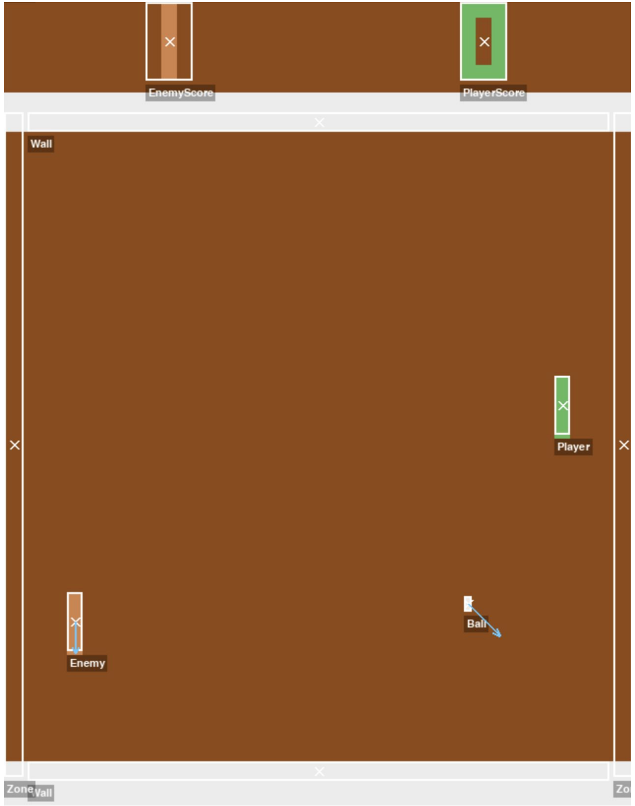
Next observation s_{t+1}

Action a

?

We want to infer $p(s_{t+1} \mid s_{1:t}, a)$

Domain – OCAtari



World modeling as an optimization problem

We can see world modeling as an optimization problem

$$p^*_{model} = \arg\min_{p_{model}} \sum_{(o_{1:T+1}, a_{1:T}) \in D} \sum_{t=1}^{T} \ell(p_{model}; o_{1:t+1}, a_{1:t})$$

where l is a loss

p_{model} could be symbolic programs or parametric neural networks

Programmatic (Symbolic) World Models

p_{model} as symbolic programs

Pros:

- Strong generalization
- Very sample efficient

Cons:

- Not flexible
    - Have only been shown to work on gridworld domains

Baseline: Direct Code Synthesis (w/ refinement)

Direct code synthesis approach

- Search in a space of symbolic programs (p_{model} are programmatic)
- Use 0-1 loss

Baseline: Direct Code Synthesis (w/ refinement)

Direct code synthesis approach

- Search in a space of symbolic programs (p_{model} are programmatic)
- Use 0-1 loss
- Use LLMs as search heuristics

Direct Code Synthesis (w/ refinement): Problems

- 2D video games, and all domains more complex than gridworlds, are very noisy, and **0-1 loss only rewards perfect prediction**
    - **Solution**: We want **stochastic world models** with log likelihood loss, even when the environment is deterministic

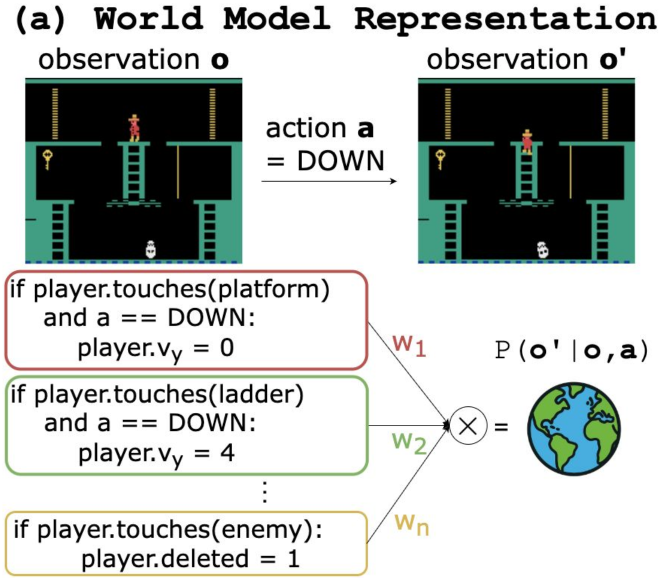
Direct Code Synthesis (w/ refinement): Problems

- 2D video games, and all domains more complex than gridworlds, are very noisy, and **0-1 loss only rewards perfect prediction**
    - **Solution**: We want **stochastic world models** with log likelihood loss, even when the environment is deterministic
- Problems with trying to find a monolithic code that explains everything
    - **Face with a big combinatorial search space**
        - Grow exponentially with the size of the programs
    - **Hard to perform targeted, local modifications**
        - With big, complex code, it’s hard to perform targeted modifications
    - **Solution:** We want **modular world models**

PoE-World

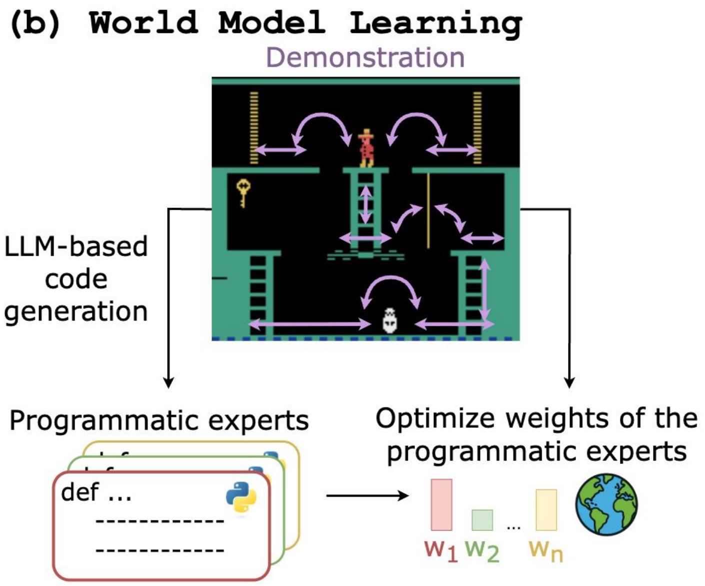
- Our solution to those problems
  - **Decompose the problem of learning a world program into learning hundreds of small programs**
    - Each of these learned programs encodes a different causal law
  - **We probabilistically aggregate to predict future observations**
    - Turning a set of pretty deterministic experts into a complex, stochastic world model

PoE-World: New representation



$$p_{\boldsymbol{\theta}}(o_{t+1}|o_{1:t}, a_{1:t}) \propto \prod_i p_i^{expert}(o_{t+1}|o_{1:t}, a_{1:t})^{\theta_i}$$

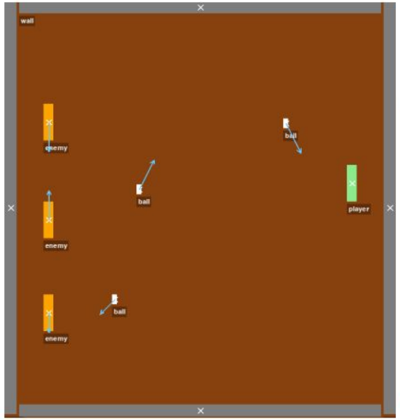
PoE-World: Learning



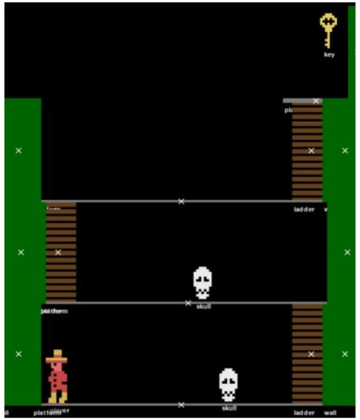
$$\theta^* = \arg\max_{\theta} \sum_{(o_{1:T+1}, a_{1:T}) \in D} \sum_{t=1}^{T} \log p_{\theta}(o_{t+1} | o_{1:t}, a_{1:t})$$

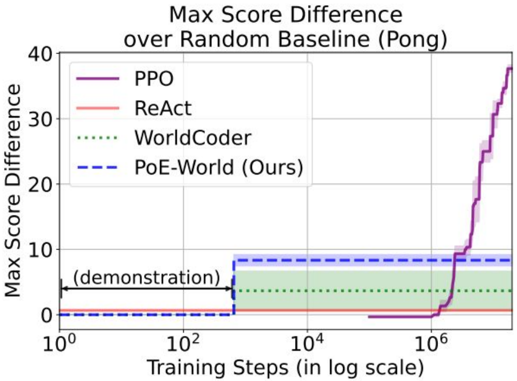
Results

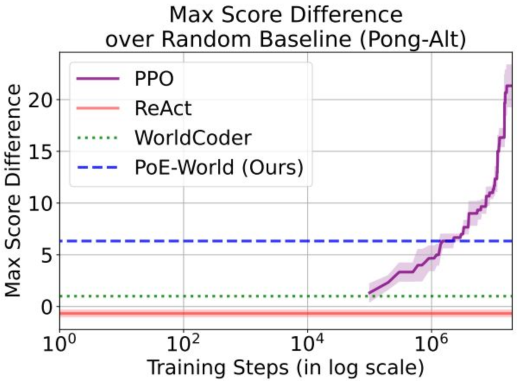
Pong-Alt

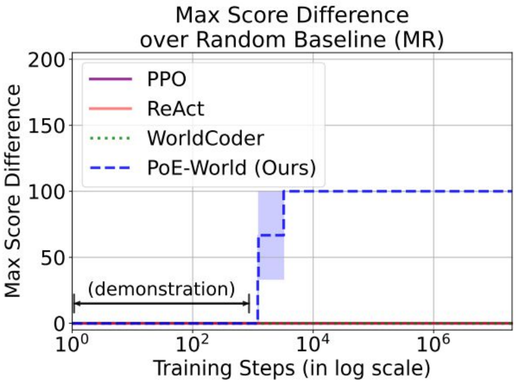


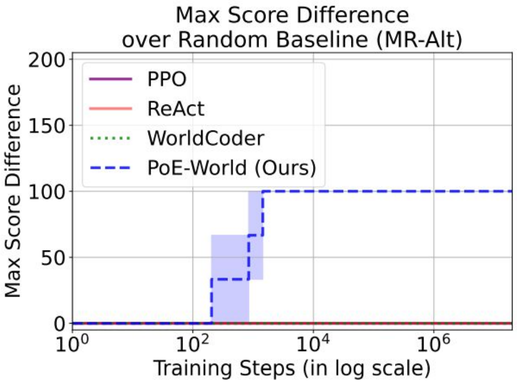
Montezuma's Revenge-Alt











Lessons Learned

- Modularity is great – makes things much more scalable
- Stochastic approximation of complex, deterministic environment is great

Lessons Learned

- Modularity is great – makes things much more scalable

- Stochastic approximation of complex, deterministic environment is great

- Low-level world models can only get us so far
  - Planning is hard with low-level world models
  - We need abstraction

- Symbolic world models can only get us so far
  - Symbolic state assumption: not everything is nice objects
  - This work relies heavily on object contacts
  - We need more flexibility to capture fine-grained details (Neurosymbolic)

Thank you!