Hi Josh,

I’m glad that both the written update and the leaderboards are providing clarity regarding the status of the project.

Following up with the last week’s update:

Learning the LossOfLife rule for breakout. We solved this issue by extrapolating objects for a longer period, which translates in the detection of the collision between the extrapolated ball and the bottom end of screen object. However, in order to avoid noise, we omitted the extrapolation of objects that collided on the timestep before they disappeared.

Damping Keyboard for Breakout and Pong. We built a controller type that can express the correct movement mechanics (alternating damped movement) for Pong and Breakout. This controller type is considered as one of the hypotheses, and we learn the relevant damping parameter. As a result of doing this, we improved our ball hitting rate in Pong from 63% to 78%.

The next thing we are going to work on is reducing noise in rule-learning, as right now we are learning many rules that associate rare conditions to effects that occur on the same timesteps.

On a separate note, in Breakout and Pong, there are cases where the ball overlaps with the paddle and the Atari game interprets that as a “graze”, whereas our VGDL simulator considers this a proper collision and makes the ball bounce off the paddle. As a result of this, the agent sometimes positions the paddle in places where it predicts a collision, but in the game itself ends up missing the ball. We concluded that this is infrequent enough that we should not address it, but rather maybe point it out in the paper as an instance of model mismatch.

I’ll upload movies from the experiments displayed on the leaderboard into a google drive folder that you can access if you are ever curious about watching those.

Best,

Juan Pablo

> For context:
> We described our difficulty learning the LossOfLife rule as follows:
> “We are having trouble learning the LossOfLife rule in Breakout, because the game environment lets us know that we’ve died long after the ball exits the screen.”
>
> This is what josh answered:
> “For learning the LossOfLife rule in Breakout, where the game environment lets you know that we’ve died long after the ball exits the screen, at least on the atari emulator I can play online, it's not a long delay... and nothing happens between the ball leaving the

screen and death registering.  Why not just make a special case rule that says when you die, assume the death event happens right after the last interaction or event observed?”

The problem:
The ball disappears from the game in breakout when it’s travelling downwards towards the bottom end of screen. The LossOfLife effect is detected around 20 steps (1/3 of a second) after the fact. We were trying to get rid of the extrapolation to reduce the noise caused by extrapolating 20 steps all objects that disappeared. The way we ended up solving this issue was by not extrapolating objects that collided with another object before they disappeared. The motivation for this is that we don’t expect an object to continue in its trajectory if it was involved in a collision right before it disappeared.

This doesn’t explain much. Can you remind me, does the ball hit the EOS and then we get a LossOfLife? Is there a lag between that collision and LossOfLife? How big is the lag? I can help streamline what you say once I know these things.