

Summer Review 7(a)

Synthesizing Programs for Images using Reinforced Adversarial Learning

Yaroslav Ganin¹, Tejas Kulkarni², Igor Babuschkin², S. M. Ali Eslami², Oriol Vinyals²

ICML 2018

Paper Link

- 1: Montreal Institute for Learning Algorithms, Montreal, Canada
2: DeepMind, London, United Kingdom.

Reviewed by : Arshdeep Sekhon

¹Department of Computer Science, University of Virginia
<https://qdata.github.io/deep2Read/>

- inverse graphics: In the visual domain, inversion of a renderer for the purposes of scene understanding
- renderer: automatic process of generating a photorealistic or non-photorealistic image from a 2D or 3D model by means of computer programs
- input programs that have sequential semantics, are composed of discrete symbols (e.g., keystrokes in a CAD program), and are long (tens or hundreds of symbols)

- an adversarially trained agent generates visual programs
- executed by a graphics engine to generate images, either conditioned on data or unconditionally
- The agent is rewarded by fooling a discriminator network
- trained with distributed reinforcement learning without any extra supervision

- target distribution of real images: p_d
- external black box renderer: input (a_1, \dots, a_n) converted to bitmap (for example a CAD program)
- $p_d \approx R(p_a)$
- p_a : policy network or agent (π RNN)
- Generation: $\pi + R$

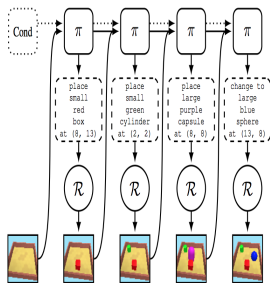


Figure: An execution trace of the SPIRAL agent. The policy outputs program fragments which are rendered into an image at each step via a graphics engine \mathcal{R} . The agent can make use of these intermediate renders to adjust its policy. The agent only receives a reward in the final step of execution

Training Objectives

Wasserstein GAN training for Discriminator:

$$L_D = -E_{x \sim p_d} D(x) + E_{x \sim p_g} D(x) + R \quad (1)$$

Generator:

$$L_G = -E_{x \sim p_g} D(x) \quad (2)$$

cannot train with naive gradient descent: black box renderer

Training generator

- maximization of the expected return which can be solved using standard techniques from reinforcement learning.
- Use Advantage Actor Critic

$$L_G = -\sum_t \pi(a_t | s_t; \theta) [R_t - V^\pi(s_t)] \quad (3)$$

- $R_t = \sum_t^N r_t$
- Optimizes 2 if

$$r_t = \begin{cases} 0, & t < N, \\ D(\mathcal{R}(a_1, a_2, \dots, a_N)), & t = N. \end{cases}$$

Figure: generator reward

Conditional generation

- condition the model on auxiliary input
- n finding a specific program that generates a given image
- supply x_{target} to the policy and to the discriminator networks.
- $p_g = R(p_a(a|x_{target}))$
- p_d becomes a Dirac δ -function centered at x_{target} .
- l_2 optimal discriminator (however, not unique)

$$-D(\mathbf{x}_{target} | \mathbf{x}_{target}) + \mathbb{E}_{\mathbf{x} \sim p_g} [D(\mathbf{x} | \mathbf{x}_{target})] .$$

Figure: generator reward

Distributed Architecture

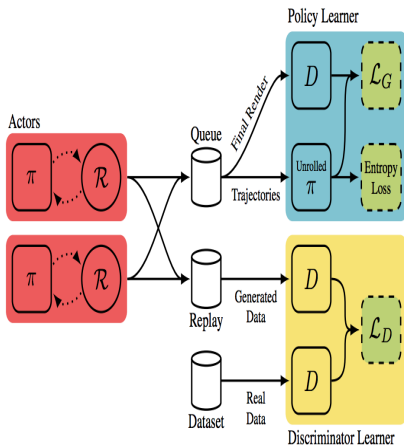
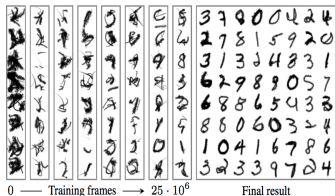


Figure: IMPALA Architecture

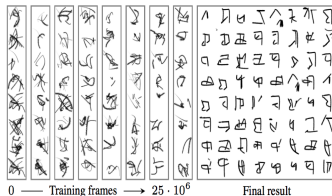
Experiments: Environments

- open-source painting library libmypaint
- The agent controls a brush and produces a sequence of (possibly disjoint) strokes on a canvas C
- The state of the environment is comprised of the contents of C as well as the current brush location l_t .
- Each action a_t is a tuple of 8 discrete decisions (a_1, a_2, \dots, a_8) [location, color, etc]
- In a 3d scene, at each time step, the agent has to decide on the object type (4 options), its location on a 16×16 grid, its size (3 options) and the color (3 color components with 4 bins each)

Results



(a) MNIST unconditional generation



(a) Omniglot unconditional generation



(b) MNIST reconstruction



(b) Omniglot reconstruction

Figure: A SPIRAL agent is trained to draw digits/alphabets via a sequence of

Results













INPUT 64×64	RECONSTRUCTION 64×64	RECONSTRUCTION 256×256	TRACE 256×256
			
			
			



Figure 10. 3D scene reconstructions. The SPIRAL agent is trained to reconstruct 3D scenes by emitting sequences of commands for the `MuJoCo` environment. In each pair, the left image corresponds to the model's output while the right one is the target. Our method is capable of accurately inferring the number of objects, their locations, sizes and colors.

Figure: