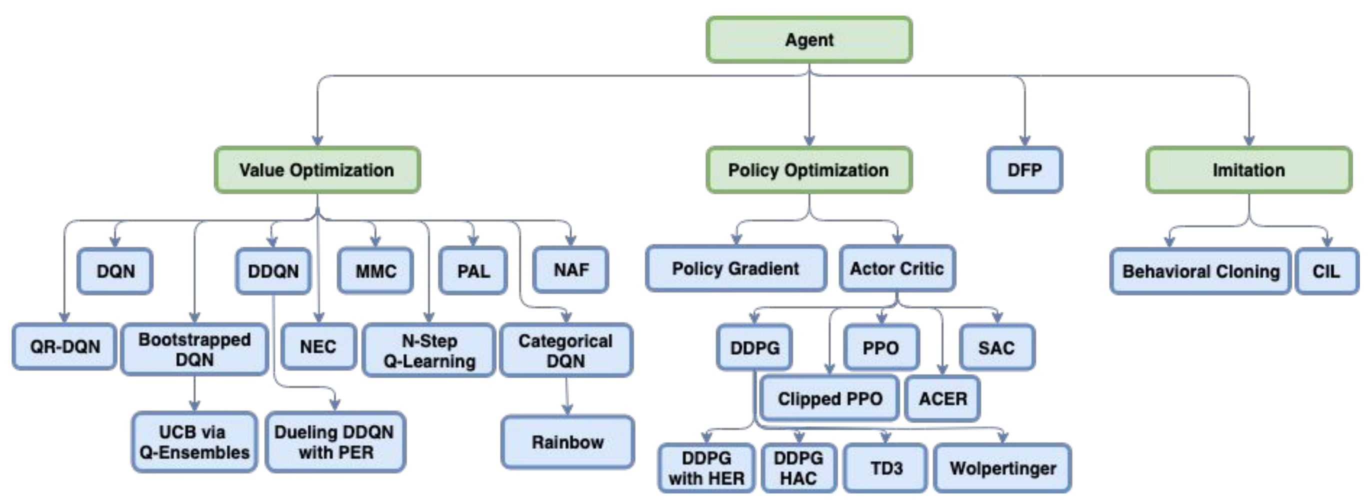
Evolving Reinforcement Learning Algorithms

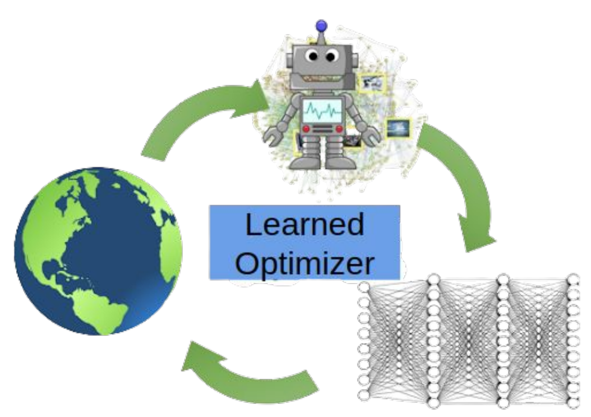
John D. Co-Reyes, Yingjie Miao, Daiyi Peng, Esteban Real, Sergey Levine, Quoc V. Le, Honglak Lee Aleksandra Faust

https://sites.google.com/view/evolvingrl

Motivation



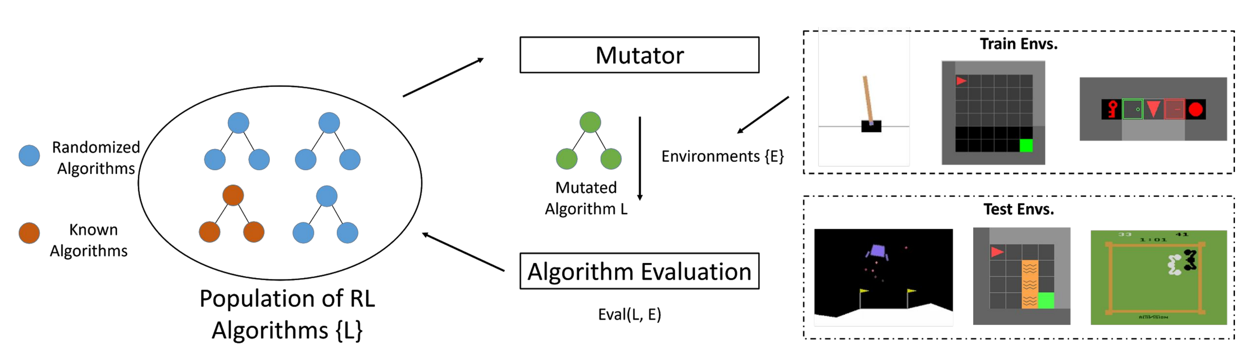
- Desire general purpose RL algorithms without manual effort.
- Problem: Meta-learn RL algorithms that generalize.



- RL algorithm as a learned optimizer
- Meta-learn the optimizer
  - Improved performance
  - Generalizable
  - Interpretable
  - Scale with compute

Overview

- **Insight:** RL algorithm as a computational graph
- **Method:** Evolve population of graphs by mutating, training, and evaluating RL agents
- **Result:** Learn new algorithms which generalize to unseen environments

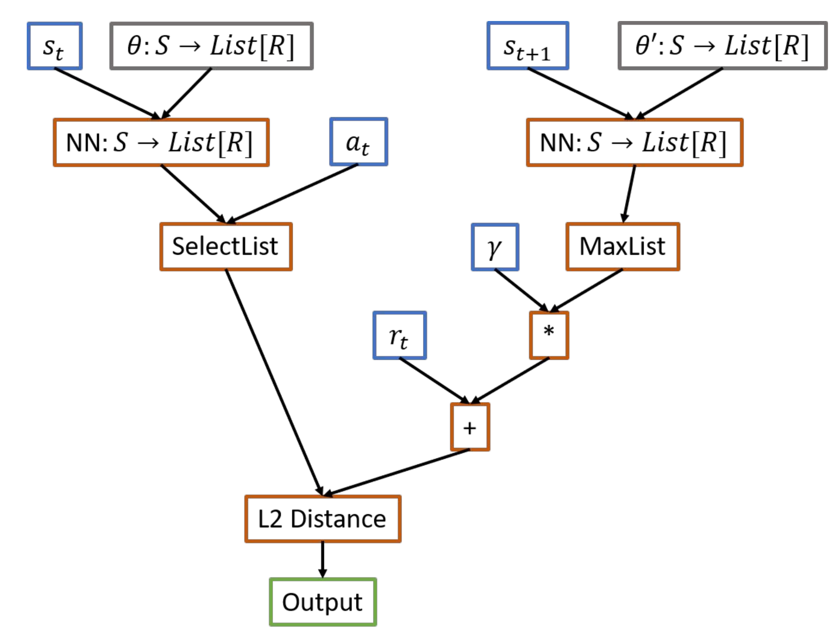


Prior Work

- **Genetic Programming**
  - Holland 1975, Koza 1993, Schmidhuber 1987
  - **AutoML**: Zoph & Le 2016, Hutter 2018, Real et al. 2020
  - Mostly applied to SL
- **Meta-learning in RL**
  - **Adaptation:** Finn & Levine 2018
  - **RNNs:** Duan et al. 2016, Wang et al. 2017
  - Not domain agnostic
- **Learning RL Algorithms**
  - **Metagradients**: Kirsch et al. 2020, Oh et al. 2020
  - Not interpretable
  - **Exploration**: Alet et al. 2020

RL Algorithm as a Computational Graph

Computational graph computes loss function for agent to optimize.



Node Types	Data Types
Inputs	State
Operators	Action
Parameters	Float
Output	List
	Probability
	Vector

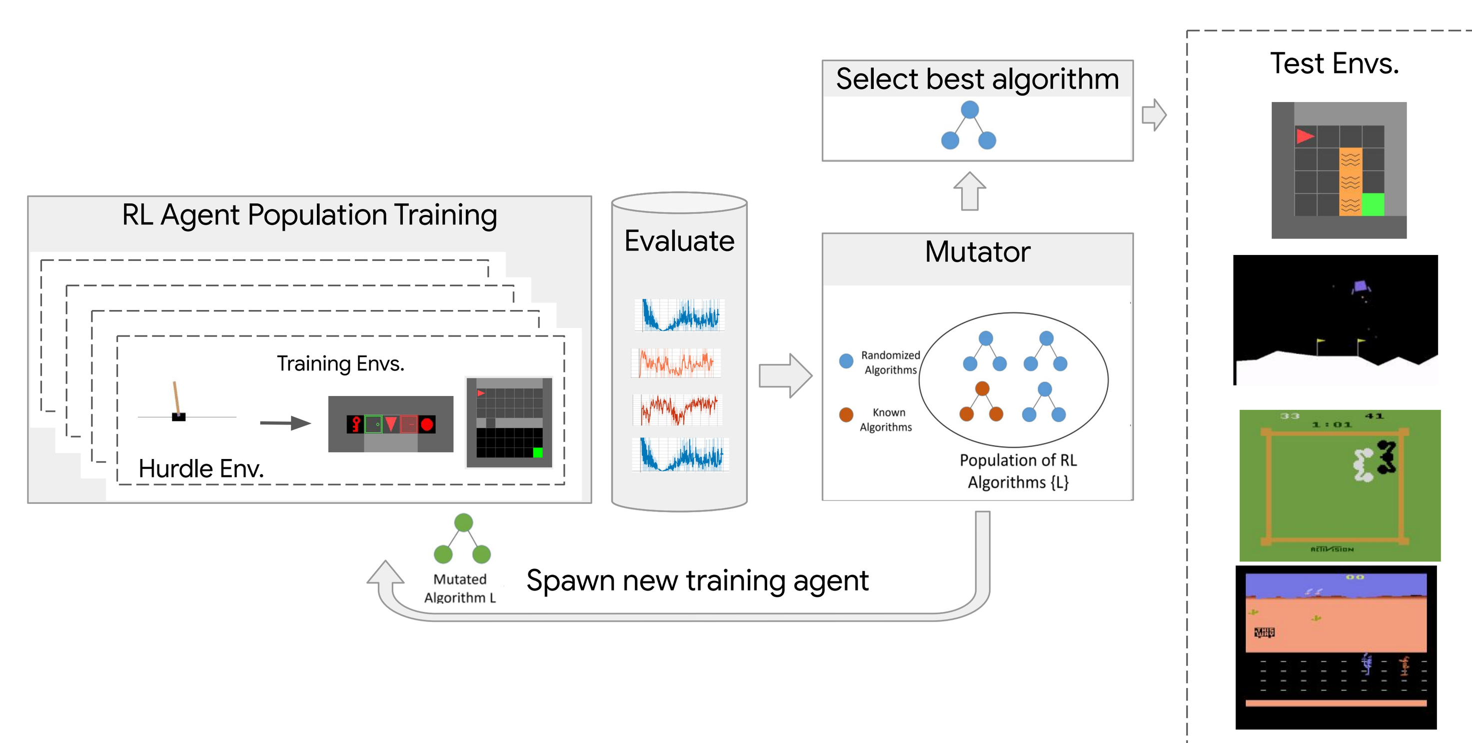
$$L_{DQN} = (Q_\theta(s_t, a_t) - (r_t + \gamma * \max_a Q_{\theta'}(s_{t+1}, a)))^2$$

Example graph for DQN loss function which computes Bellman squared error using two Q value networks.

- Directed acyclic graph computes DQN style loss function for value-based agent
- Node types include neural network operators to support more complex architectures
- Data types allows for type checking and ruling out invalid graphs
- Functional equivalence checker skips graphs that are functionally the same
- Representation is expressive, interpretable, and generalizable

Operation	Input Types	Output Type
Add	$\mathbb{X}, \mathbb{X}$	$\mathbb{X}$
Subtract	$\mathbb{X}, \mathbb{X}$	$\mathbb{X}$
Max	$\mathbb{X}, \mathbb{X}$	$\mathbb{X}$
Min	$\mathbb{X}, \mathbb{X}$	$\mathbb{X}$
DotProduct	$\mathbb{X}, \mathbb{X}$	$\mathbb{R}$
Div	$\mathbb{X}, \mathbb{X}$	$\mathbb{X}$
L2Distance	$\mathbb{X}, \mathbb{X}$	$\mathbb{R}$
MaxList	$List[\mathbb{R}]$	$\mathbb{R}$
MinList	$List[\mathbb{R}]$	$\mathbb{R}$
ArgMaxList	$List[\mathbb{R}]$	$\mathbb{Z}$
SelectList	$List[\mathbb{X}], \mathbb{Z}$	$\mathbb{X}$
MeanList	$List[\mathbb{X}]$	$\mathbb{X}$
VarianceList	$List[\mathbb{X}]$	$\mathbb{X}$
Log	$\mathbb{X}$	$\mathbb{X}$
Exp	$\mathbb{X}$	$\mathbb{X}$
Abs	$\mathbb{X}$	$\mathbb{X}$
(C)NN:$\mathbb{S} \to List[\mathbb{R}]$	$\mathbb{S}$	$List[\mathbb{R}]$
(C)NN:$\mathbb{S} \to \mathbb{R}$	$\mathbb{S}$	$\mathbb{R}$
(C)NN:$\mathbb{S} \to \mathbb{V}$	$\mathbb{V}$	$\mathbb{V}$
Softmax	$List[\mathbb{R}]$	$\mathbb{P}$
KLDiv	$\mathbb{P}, \mathbb{P}$	$\mathbb{R}$
Entropy	$\mathbb{P}$	$\mathbb{R}$
Constant		1, 0.5, 0.2, 0.1, 0.01
MultiplyTenth	$\mathbb{X}$	$\mathbb{X}$
Normal(0, 1)		$\mathbb{R}$
Uniform(0, 1)		$\mathbb{R}$

Evolve Population of RL Algorithms



- Initialize population of ~ 300 RL algorithms with randomized computation graphs
- Evaluate performance by training over set of diverse but cheap envs.
- Mutate most promising algorithms to spawn new agents for evaluation
- Regularized Evolution removes oldest algorithms from population
- Hurdle env. stops bad algorithms early
- Can bootstrap from existing algorithms
- Evolutionary method can scale with compute

Learned Algorithms

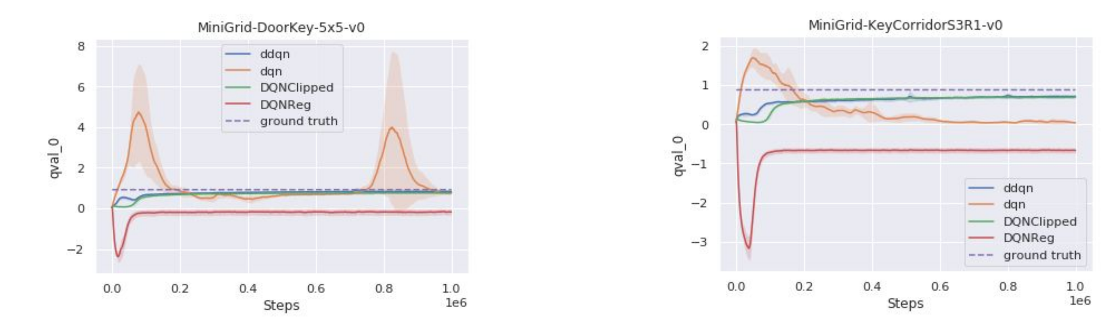
$$Y_t = r_t + \gamma * \max_a Q_{targ}(s_t, a), \text{ and } \delta = Q(s_t, a_t) - Y_t.$$

$$L_{\text{DQNClipped}} = \max\left[Q(s_t, a_t), \delta^2 + Y_t\right] + \max\left[Q(s_t, a_t) - Y_t, \gamma(\max_a Q_{targ}(s_t, a))^2\right]$$

- DQNClipped as constrained optimization

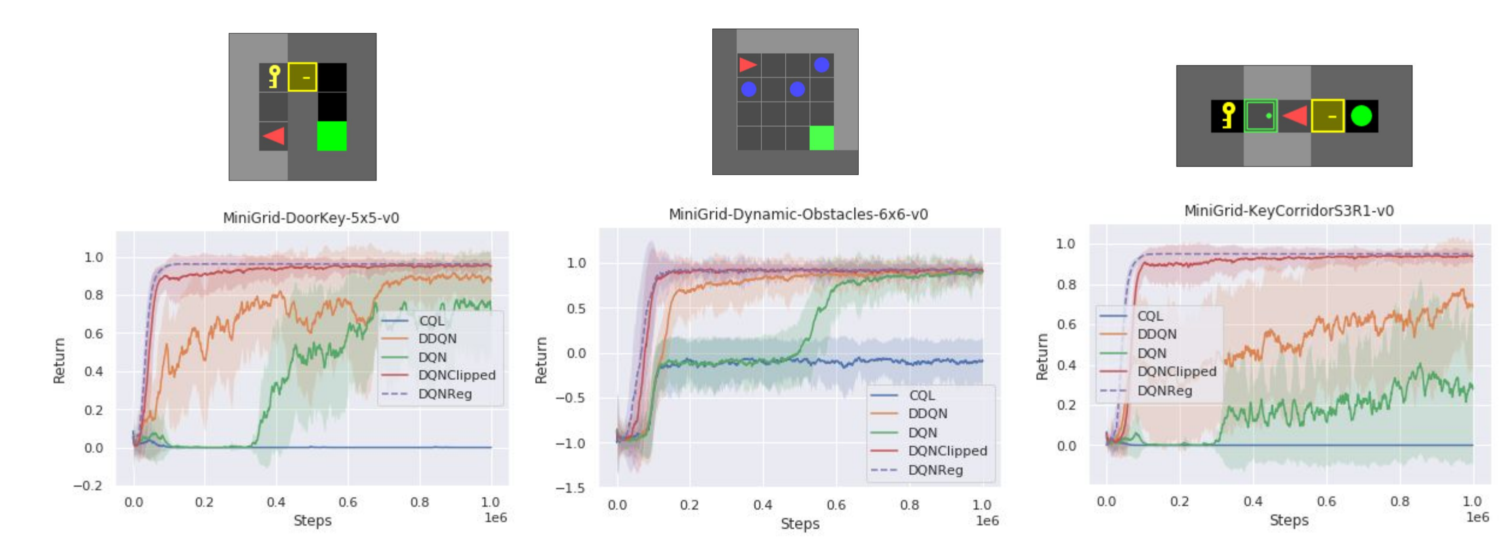
$$L_{\text{DQNReg}} = 0.1 * Q(s_t, a_t) + \delta^2$$

- DQNReg as entropy regularization

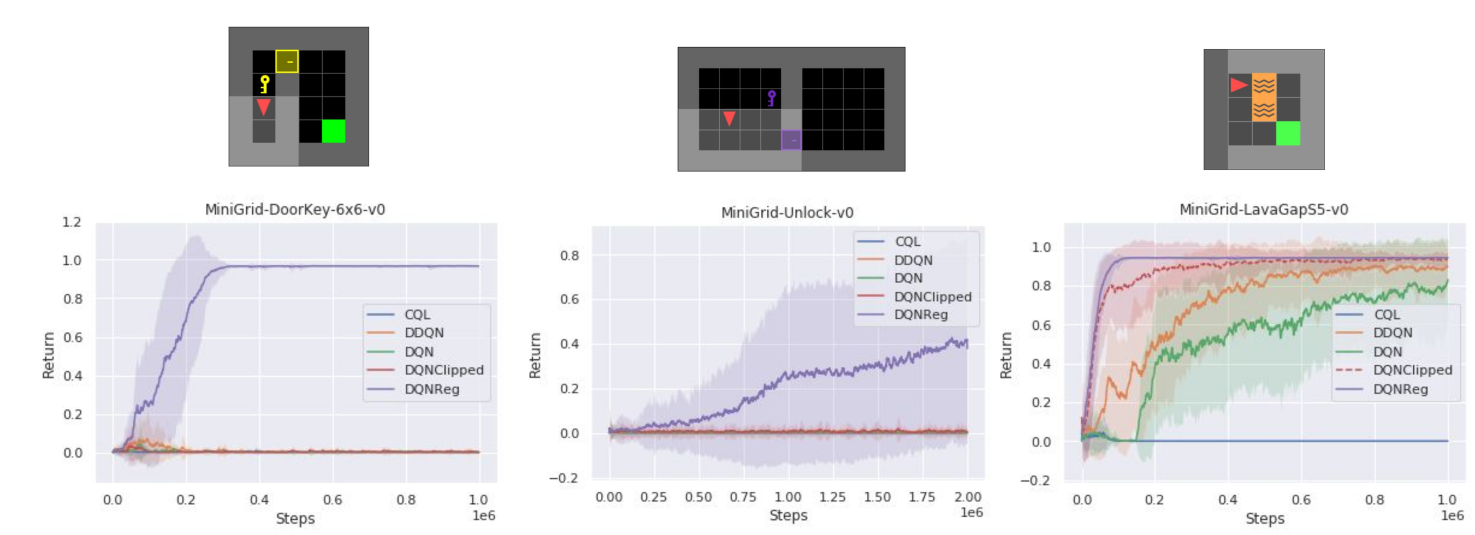


- Learned algorithms prevent value overestimation in different ways

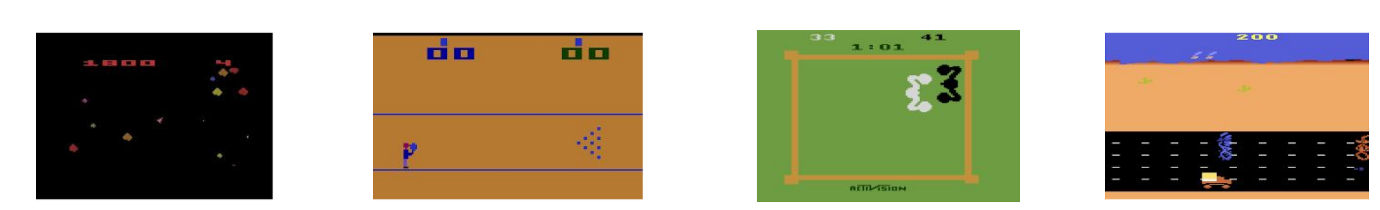
Results



- Outperform baselines on train envs.



- Generalize to unseen environments.



Env	DQN	DDQN	PPO	DQNReg
Asteroid	1364.5	734.7	2097.5	**2390.4**
Bowling	50.4	68.1	40.1	**80.5**
Boxing	88.0	91.6	94.6	**100.0**
RoadRunner	39544.0	44127.0	35466.0	**65516.0**

- Benefits on Atari even though training envs. were non-image based

Future Work

- Extensions to actor critic, offline RL, representation learning
- Analyze and incorporate learned algorithms into existing ones
- Machine assisted algorithm development