Automated Reinforcement Learning (AutoRL): A Survey and Open Problems

(Jan 2022, Facebook AI etc..)

Introduction

RL is sensitive about hyp-param, training process, design choices

AutoML has shown it is possible to automate in supervised learning

But, AutoRL(AutoML+RL) is challenging. Because of **non-stationarity**, **diversity of methods**.

*Non-stationarity : with trial-and-error, agent (re-)generates its own training data, which is highly dependent on the instantiation of the agent(even in stationary environment) and only useful of current policy

Goal : present the field of AutoRL. taxonomy(분류), pipeline

What needs to be Automated?

Task：Reward function, Observation, Action space

Algorithm : DQN(Atari), SAC(MuJoCo) => meta-learner(RNN), learn objective function itself

Architecture : proprioceptive state(MLP), pixels(IMPALA-CNN)

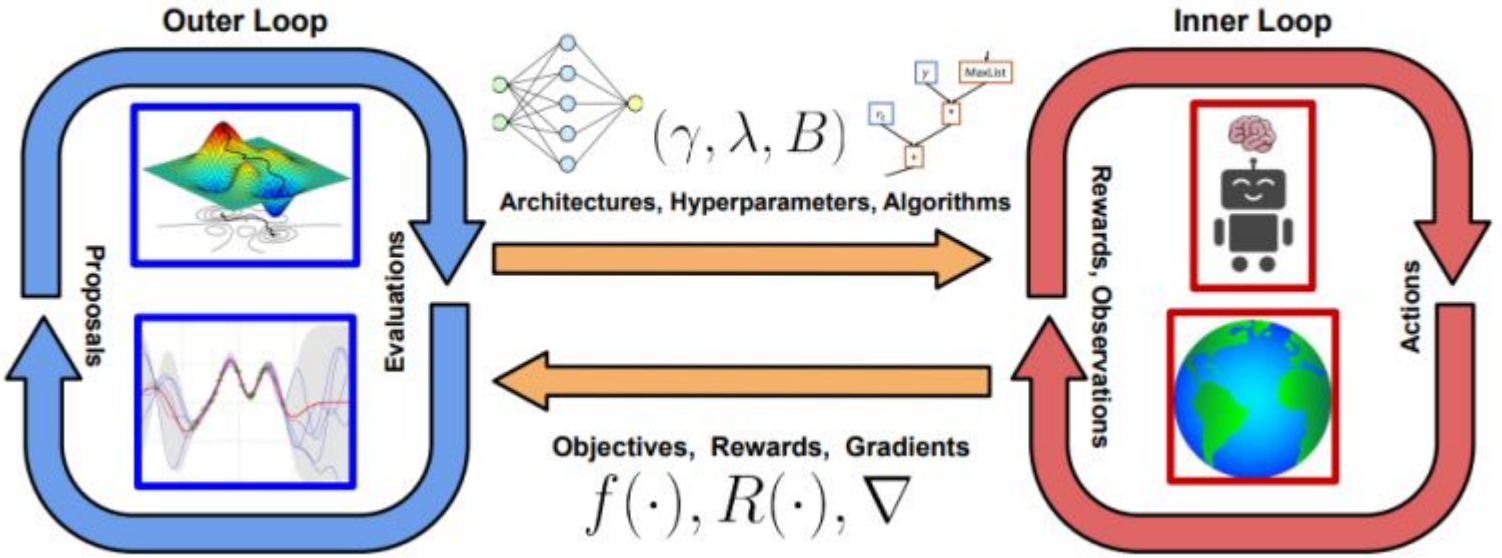
Hyper params : discount factor, step-size(TD), eligibility trace parameter, Loss(MSE or Huber), N-step return, regularizer, reward normalization and seed

But may be suboptimal without vast resource…

Notation

$\mathcal{S}$	State space
$\mathcal{A}$	Action space
P	Transition function
R	Reward function
$\mathcal{O}$	Observation space
ρ_o	Initial state distribution
T	Set of terminal states
γ	Discount factor
V	State-value function
Q	Action-value function
J	Expected total reward function
L	Loss function
f	Objective function
θ	Neural network parameters
ϕ	Secondary neural network parameters
B	Batch size
λ	Bootstrapping hyperparameter
ζ	All hyperparameters
η	Subset of hyperparameters that are differentiable

Inner loop & Outer loop



Inner loop : standard RL pipeline

Outer loop : optimize the agent configuration

High level summary of AutoRL

Class	Algorithm properties				What is automated?
Random/Grid Search (4.1)	††††	■	⇉	✓	hyperparameters, architecture, algorithm
Bayesian Optimization (4.2)	††††	■	⇉	✓	hyperparameters, architecture, algorithm
Evolutionary Approaches (4.3)	††††	■	⇉	✓	hyperparameters, architecture, algorithm
Population Based Training (4.4)	††††	■	⇉	✓	hyperparameters, architecture
Meta-Gradients (4.5)	†	∇	→	●	hyperparameters
Blackbox Online Tuning (4.6)	†	■	→	✓	hyperparameters
Learning Algorithms (4.7)	††††	■	⇉	●	algorithm
Environment Design (4.8)	††††	■	⇉	●	environment

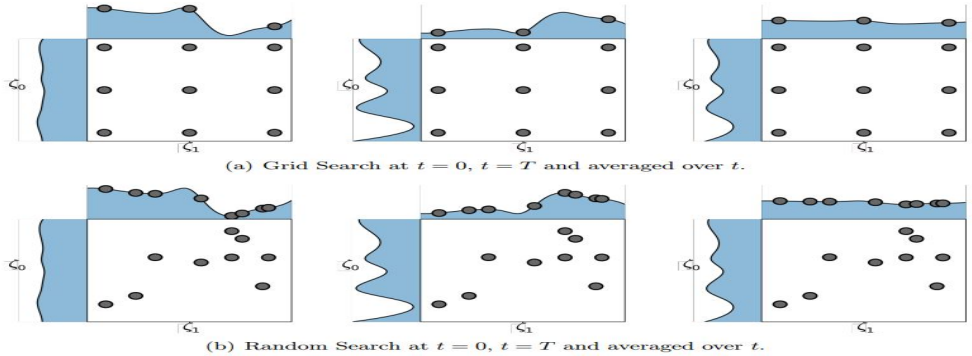
† only uses a single trial, †††† requires multiple trials

∇ requires differentiable variables, ■ works with non-differentiable hyperparameters

⇉ parallelizable → not parallelizable

✓ works for any RL algorithm, ● works for only some classes of RL algorithms,

1.Random/Grid Search



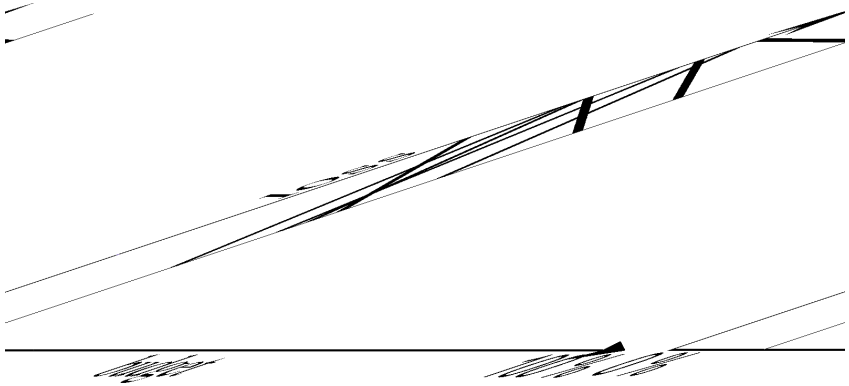
(a) Grid Search at $t = 0$, $t = T$ and averaged over t.

(b) Random Search at $t = 0$, $t = T$ and averaged over t.

Pros : easy, start on fly

Cons : can't solve non-stationarity, sub-optimal(unable to leverage the information regarding promising regions, which hyp is important), only explorate not exploit, poorly as search space increases.

Ex. Hyperband (Li, 2017), Hyperband uses Successive Halving (Jamieson & Talwalkar, 2016), Zhang et al.(2021)



*Successive halving : give the most budget to the most promising methods
*Hyper band : successive halving with different budgets

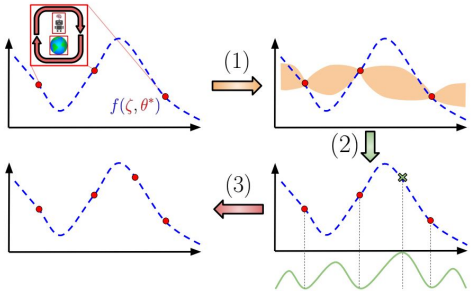
2. Bayesian Optimization(BO)

learning based on previous observation

Pros : leveraging different fidelities(e.g. seed, epoch, internal information)

Cons : big overhead, need expensice black-box evaluation(check accuracy in the end) -> leveraging different fidelities(e.g. seed, epoch, internal information), static tuning

Ex. BOIL(Nguyen,2020), Hetel et al(2020)



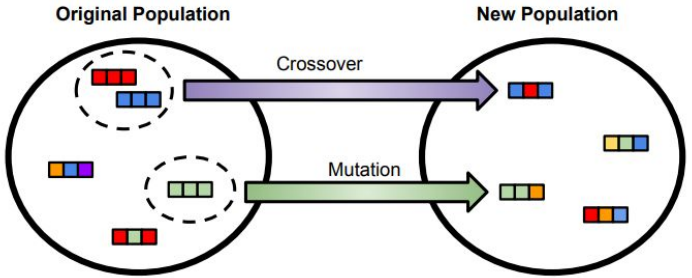
Bayesian Optimization, which consists of 3 main steps. (1): A regressor (commonly a Gaussian Process) is used to construct uncertainty estimates on the blackbox function $f(\cdot)$ given previous evaluation data. (2): An acquisition function is constructed from the regressor, which represents the explore-exploit tradeoff on f. (3): The argmax of the acquisition function is used for the next trial.

3.Evolutionary Approaches

Pros : weights of archtectures evolution(neuro evolution), hyp tuning, escape from local minima by automatic restarting strategies(1998), DDPG with HER(2021)

Cons : inefficiency, need CPU

Ex. NEAT(2002), HyperNEAT(2009), WANNs(2019), GA(Michalewicz, 2013), WOW(2021, 혹등고래 사냥전략?)



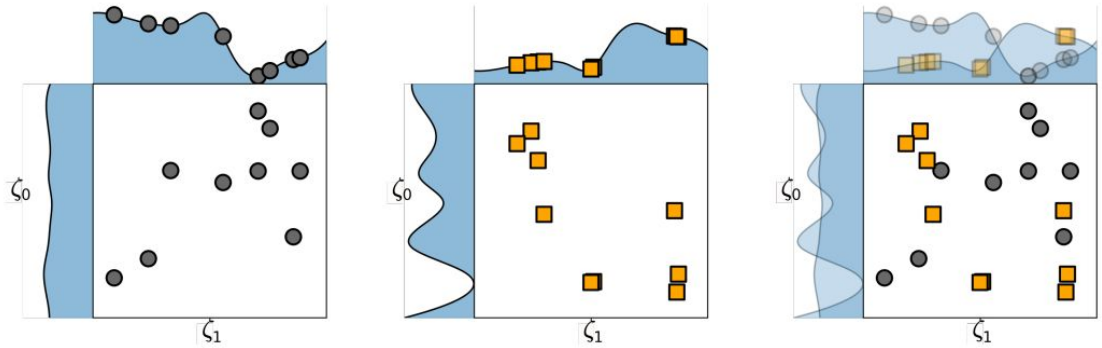
4.Population Based Training(PBT)

weak agent replaced with strong agent!

Pros : train multiple angets(+hyp search) in parallel, learn hyp schedules(effective in non-stationarity), hyp fly

Cons : data inefiiciency, little research understanding meta-parameter(population size)

Ex. SEARL(2021), PBT-BT(2021), Dalibard et al(2021)



5.Meta-Gradients for Online Tuning

hyp의 subset을 meta-parameter로 정의하여, inner-loop, outer-loop으로 나눠, 최적화

Pros : dealing with non-stationarity of RL hyp. learn non-trivial hyp schedules, good at Atari,Robot tasks. discover auxiliary tasks. learn RL objectives online, stron asymptotic performance. hyp on fly

Cons : rely on meta-params initialization. non-differentiable hyp에는 적용불가(e.g. choice of optimizer, activation function).

Ex. STAC, STACX(Zahavy,2020), Bootstrapped Meta-learning(Flennerhag, 2021),FRODO(Zhongwen, 2020), DARTS(Liu, 2019a)

6.Blackbox Online Tuning

Pros : hyp on fly, ok with non-differentiable hyp

Cons : single agent, limited scope for the search space(bandit algorithm must be able to explore all arms), assuming all arms to be independents is decrease efficiency

Ex. Sutton(1994), Kearns(2000), Downey(2010), White(2016), HOOF(Paul,2019), Scahul(2020)->Agent57, …

7.Learning Reinforcement Learning Algorithm

learning to learn

Pros : learn RL tasks using RNN agent, automating design of RL loss function

- **Neral loss function** : nn with parameters ζ optimized via ES or gradient…
- **Symbolic loss function** : represented as a symbolic expression consisting of predefined primitives, akin to genetic programming

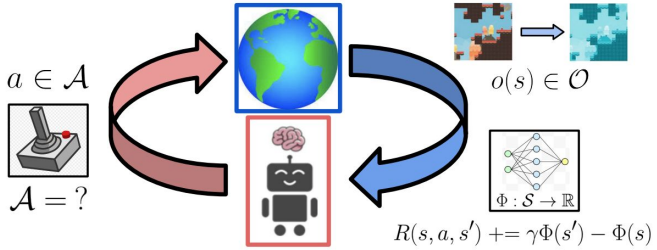
Cons : difficult to understand analytical properties of neural loss functions, need explaining why they work. Unclear what the optimal design choices are.

Ex. Wang(2016), PEARL(Rakelly, 2019), variBAD(Zintgraf, 2020)

Ex of nlf. Bechtle(2020), LPG(Oh, 2020)

Ex of slf. Alet(2020), Co-Rtes(2021), Laroche(2018)

8.Environment Design



auto R, A, O?
design env to faster training

Curriculum learning ~ syntheic env learning ~ generation

Pros :

- R(bi-level optimization, intrinsic or extrinsic)
- A(simplifying, repeat one action or construct new action space)
- O(augmentation)

Cons : Unclear which of them leads to the biggest gains in performance.

ex of R : Zheng(2018), Hu(2020), Zou(2019), …, Faust(2019)

ex of A : Sharma(2017), Bidenkapp(2021), Bacon(2017), Mankowitz(2018)

ex or O : Raileanu(2020)

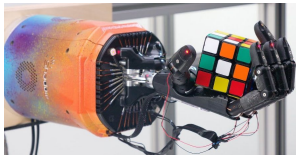
8-1.Multiple Components of Env.. UL or SL??

Uunsupervised

- curriculum learning(modify state space S, and initial distribution ρ)
- encoding env. 랭킹변화시 random mutation
- regret, skill level에 따라 적절한 환경 제공

Ex. POET(Wang,2019,2020), Lee(2020), PAIRED(Dennis, 2020), CoDE(Gur,2021), PLR(Jiang,2021), Bontrager(2020)

Supervised



- 간단한거 여러개를 풀게해서 어려운거 풀기
- 환경의 난이도 추정이 쉬움
- GAN을 활용해 agent가 학습할수록 더 어려운 task 만들어냄

Ex. OpenAI(2019), Klink(2020), Wphlke(2020), Florensa(2018)~, da Silva(2019), AMIGo(Campero 2021), APT-Gen(Fang,2021)

8.Environment Design

auto R, A, O?
design env to faster training

Curriculum learning ~ syntheic env learning ~ generation

Pros :

- R(bi-level optimization, intrinsic or extrinsic)
- A(simplifying, repeat one action or construct new action space)
- O(augmentation)

Cons : Unclear which of them leads to the biggest gains in performance.

ex of R : Zheng(2018), Hu(2020), Zou(2019), …, Faust(2019)

ex of A : Sharma(2017), Bidenkapp(2021), Bacon(2017), Mankowitz(2018)

ex or O : Raileanu(2020)

8.Environment Design

auto R, A, O?
design env to faster training

Curriculum learning ~ syntheic env learning ~ generation

Pros :

- R(bi-level optimization, intrinsic or extrinsic)
- A(simplifying, repeat one action or construct new action space)
- O(augmentation)

Cons : Unclear which of them leads to the biggest gains in performance.

ex of R : Zheng(2018), Hu(2020), Zou(2019), …, Faust(2019)

ex of A : Sharma(2017), Bidenkapp(2021), Bacon(2017), Mankowitz(2018)

ex or O : Raileanu(2020)

9. Hybrid Approaches

이도저도 아니거나, 위에서 말한 방법론 같이 쓴거

BOHB(2018, Hyperband + BO)

DEHB(2021, Hyperband + Differential Evolution)

…

PB2(2020, PBT + BO)

Pros : 병렬자원에서 효율적, 꽤나 유명함(RNA design, human level in Quake 3)

Cons : 어떻게 섞어야 best인지 모름. 알아내는 것도 힘듬

10. Conclusion

Introduced AutoRL by discussing a wide variety of methods to automate the RL training pipeline.

SL : open-loop, one-step

RL : closed-loop, noisy (Challenge!)

연구하기엔 딱좋은 분야일수도?

Reference

https://neptune.ai/blog/hyperband-and-bohb-understanding-state-of-the-art-hyperparameter-optimization-algorithms (BO, Hyper Band, BOHB)

https://blog.naver.com/ehdrndd/222627248292 (Blog)