Offline Reinforcement Learning as One Big Sequence Modeling Problem

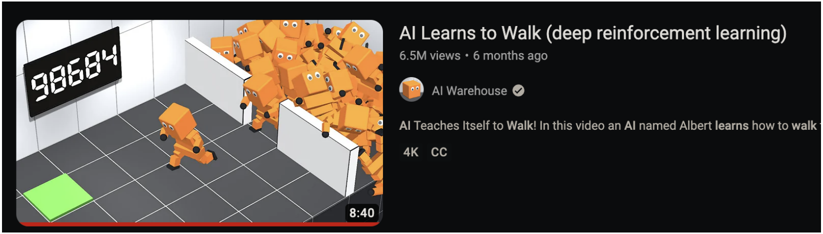
Michael Janner, Qiyang Li, & Sergey Levine 2021

Elliot Hill

Duke University

Oct, 2023

Reinforcement learning (RL) provides a mathematical formalism for learning-based control



AI Learns to Walk (deep reinforcement learning)

6.5M views • 6 months ago

AI Warehouse

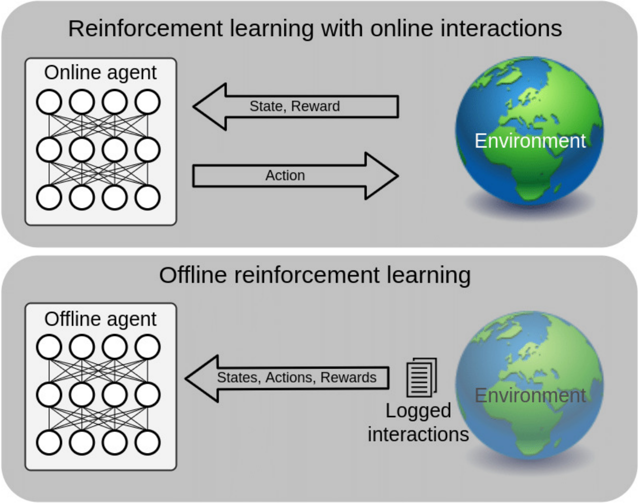
AI Teaches Itself to **Walk**! In this video an **AI** named Albert **learns** how to **walk**

4K CC

RL is a framework to build decision-making agents

Agents aim to learn optimal behavior (policy) by interacting with the environment through trial and error and receiving rewards as feedback

Offline RL is closer to traditional ML than online RL



de Lima and Krohling [2021]

RL is a group of fields that developed independently working on different types of problems and domains

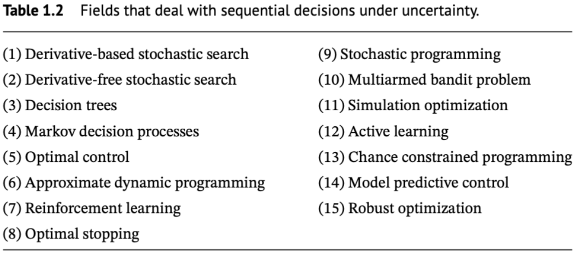
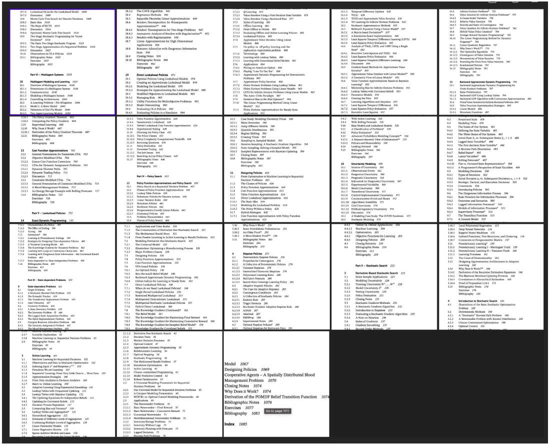


Table 1.2 Fields that deal with sequential decisions under uncertainty.

(1) Derivative-based stochastic search
(2) Derivative-free stochastic search
(3) Decision trees
(4) Markov decision processes
(5) Optimal control
(6) Approximate dynamic programming
(7) Reinforcement learning
(8) Optimal stopping
(9) Stochastic programming
(10) Multiarmed bandit problem
(11) Simulation optimization
(12) Active learning
(13) Chance constrained programming
(14) Model predictive control
(15) Robust optimization

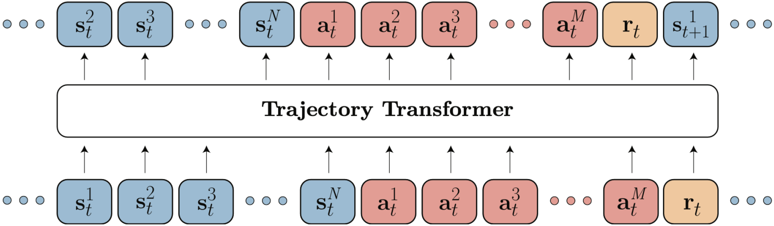
Powell [2022]

RL subfields develop specific approaches for solving their problem of interest



Powell [2022]

Big idea: rather than use specific RL methods to solve a problem, translate the problem into a general sequence modeling problem, then solve with standard methods


$\mathbf{s}_t^2$ $\mathbf{s}_t^3$ $\mathbf{s}_t^N$ $\mathbf{a}_t^1$ $\mathbf{a}_t^2$ $\mathbf{a}_t^3$ $\mathbf{a}_t^M$ $\mathbf{r}_t$ $\mathbf{s}_{t+1}^1$
Trajectory Transformer
$\mathbf{s}_t^1$ $\mathbf{s}_t^2$ $\mathbf{s}_t^3$ $\mathbf{s}_t^N$ $\mathbf{a}_t^1$ $\mathbf{a}_t^2$ $\mathbf{a}_t^3$ $\mathbf{a}_t^M$ $\mathbf{r}_t$

A trajectory sequence τ consists of T states (s), actions (a), and scalar rewards (r)

$$\tau = (s_1, a_1, r_1, s_2, a_2, r_2, \ldots, s_T, a_T, r_T)$$

In the case of continuous states and actions, they discretize each dimension (either uniformly or by quantile)

$$\tau = (\ldots, s_t^1, s_t^2, \ldots, s_t^N, a_t^1, a_t^2, \ldots, a_t^M, r_t, \ldots) \quad t = 1, \ldots, T.$$

Token subscripts denote timestep and superscripts on states and actions denote dimension (i.e., s_t^i is the ith dimension of the state at time t)

Beam search is used to decode sequences

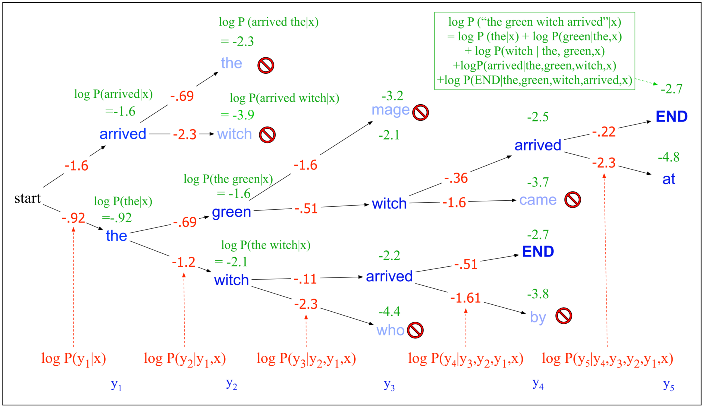
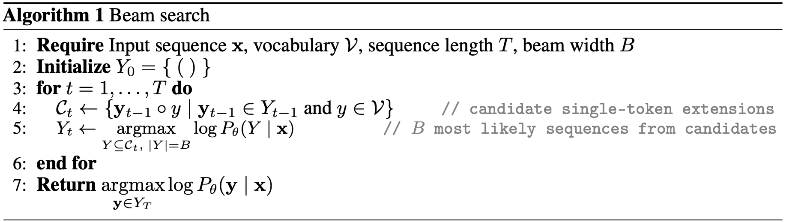


Figure 11.13 Scoring for beam search decoding with a beam width of $k = 2$. We maintain the log probability of each hypothesis in the beam by incrementally adding the logprob of generating each next token. Only the top k paths are extended to the next step.

Jurafsky [2000]

Beam search is used to decode sequences



Algorithm 1 Beam search

```
Require Input sequence $\mathbf{x}$, vocabulary $\mathcal{V}$, sequence length $T$, beam width $B$
Initialize $Y_0 = \{\ ()\ \}$
for $t = 1, \ldots, T$ do
    $\mathcal{C}_t \leftarrow \{\mathbf{y}_{t-1} \circ y \mid \mathbf{y}_{t-1} \in Y_{t-1}$ and $y \in \mathcal{V}\}$    // candidate single-token extensions
    $Y_t \leftarrow \underset{Y \subseteq \mathcal{C}_t,\ |Y|=B}{\operatorname{argmax}} \log P_\theta(Y \mid \mathbf{x})$    // $B$ most likely sequences from candidates
end for
Return $\underset{\mathbf{y} \in Y_T}{\operatorname{argmax}} \log P_\theta(\mathbf{y} \mid \mathbf{x})$
```

Three types of problems are tested: imitation learning, goal-conditioned RL, and offline RL

Imitation learning

The goal is to reproduce the distribution of trajectories in the training data, we can optimize directly for the probability of a trajectory τ

Goal-conditioned reinforcement learning

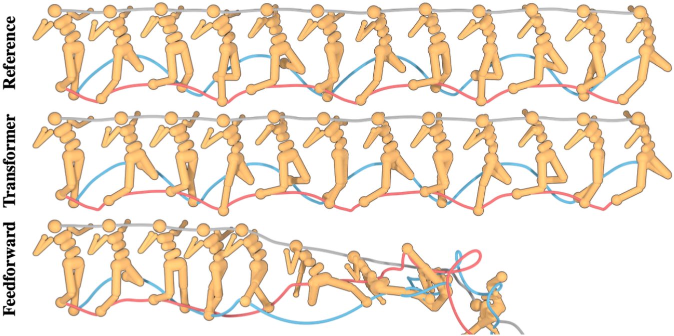
Trains an agent to achieve different goals under particular scenarios; i.e., we want the reach some desired final state s_T

Offline reinforcement learning

The agent learns a policy to optimize the predicted reward signal rather than the log probability of a sequence

Imitation learning

Imitation learning matches the goal of sequence modeling exactly, so we can use beam search without modification by setting the conditioning input x to the current state s_t



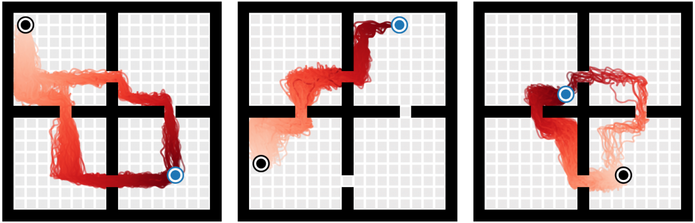
Goal-conditioned reinforcement learning

An attention mask is used to ensure that predictions only depend on the previous tokens in a sequence, disallowing future events to affect the past

$$P_\theta(s_t^i | s_t^{<i}, \tau_{<t}, s_T)$$

We can use this directly as a goal-reaching method by conditioning on the final state s_T (i.e., append s_T to the input sequence)

$$s_T, s_1, s_2, \ldots, s_{T-1}$$



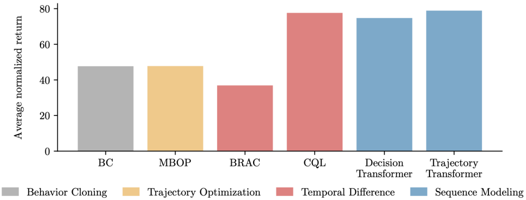
Offline reinforcement learning

Beam search optimizes sequences for their probability under the data distribution.

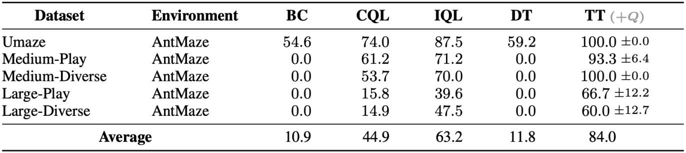
In the offline RL, they replace the log-probabilities of transitions with the predicted reward signal (log-probability of optimality) so that the Trajectory Transformer can be used for reward-maximizing behavior.

The Trajectory Transformer is competitive with the best prior problem-specific offline-RL methods

Dataset	Environment	BC	MBOP	BRAC	CQL	DT	TT (uniform)	TT (quantile)
Med-Expert	HalfCheetah	59.9	105.9	41.9	91.6	86.8	40.8 ±2.3	95.0 ±0.2
Med-Expert	Hopper	79.6	55.1	0.9	105.4	107.6	106.0 ±0.28	110.0 ±2.7
Med-Expert	Walker2d	36.6	70.2	81.6	108.8	108.1	91.0 ±2.8	101.9 ±6.8
Medium	HalfCheetah	43.1	44.6	46.3	44.0	42.6	44.0 ±0.31	46.9 ±0.4
Medium	Hopper	63.9	48.8	31.3	58.5	67.6	67.4 ±2.9	61.1 ±3.6
Medium	Walker2d	77.3	41.0	81.1	72.5	74.0	81.3 ±2.1	79.0 ±2.8
Med-Replay	HalfCheetah	4.3	42.3	47.7	45.5	36.6	44.1 ±0.9	41.9 ±2.5
Med-Replay	Hopper	27.6	12.4	0.6	95.0	82.7	99.4 ±3.2	91.5 ±3.6
Med-Replay	Walker2d	36.9	9.7	0.9	77.2	66.6	79.4 ±3.3	82.6 ±6.9
Average		47.7	47.8	36.9	77.6	74.7	72.6	78.9



The Trajectory Transformer performs even better when combined with traditional RL methods



Dataset	Environment	BC	CQL	IQL	DT	TT (+Q)
Umaze	AntMaze	54.6	74.0	87.5	59.2	100.0 ±0.0
Medium-Play	AntMaze	0.0	61.2	71.2	0.0	93.3 ±6.4
Medium-Diverse	AntMaze	0.0	53.7	70.0	0.0	100.0 ±0.0
Large-Play	AntMaze	0.0	15.8	39.6	0.0	66.7 ±12.2
Large-Diverse	AntMaze	0.0	14.9	47.5	0.0	60.0 ±12.7
Average		10.9	44.9	63.2	11.8	84.0

Summary

The algorithm trains a sequence model jointly on states, actions, and rewards and samples from it using beam search.

By reframing RL as a sequence modeling problem, you can ignore many of the complications of RL (e.g., policies, value functions).

Is it worth read?
Yes, if you are interested in RL. It recasts RL problems in a novel way that will also feel familiar if you have done any sequence modeling.

Is it worth implementing?
Definitely. All methods used in the paper are standard and readily available sequence modeling methods.

References

Leandro de Lima and Renato Krohling. Discovering an aid policy to minimize student evasion using offline reinforcement learning, 04 2021.

Michael Janner, Qiyang Li, and Sergey Levine. Offline reinforcement learning as one big sequence modeling problem. *Advances in neural information processing systems*, 34:1273–1286, 2021.

Dan Jurafsky. *Speech & language processing*. Pearson Education India, 2000.

Warren B Powell. *Reinforcement Learning and Stochastic Optimization: A unified framework for sequential decisions*. John Wiley & Sons, 2022.