Reinforcement Learning for Personalized Drug Discovery and Design for Complex Diseases: A Systems Pharmacology Perspective

Ryan K. Tan[1], Yang Liu[1], Lei Xie[1,2,3,*]

[1]Department of Computer Science, Hunter College, The City University of New York

[2]Ph.D. Program in Computer Science, Biology & Biochemistry, The Graduate Center, The City University of New York

[3]Helen and Robert Appel Alzheimer's Disease Research Institute, Feil Family Brain & Mind Research Institute, Weill Cornell Medicine, Cornell University

*Correspondence should be addressed

Abstract

Many multi-genic systematic diseases such as Alzheimer's disease and majority of cancers do not have effective treatments yet. Systems pharmacology is a potentially effective approach to designing personalized therapies for untreatable complexed diseases. In this article, we review the potential of reinforcement learning in systems pharmacology-oriented drug discovery and design. In spite of successful application of advanced reinforcement learning techniques to target-based drug discovery, new reinforcement learning techniques are needed to boost generalizability and transferability of reinforcement learning in partially observed and changing environments, optimize multi-objective reward functions for system-level molecular phenotype readouts and generalize predictive models for out-of-distribution data. A synergistic integration of reinforcement learning with other machine learning techniques and related fields such as biophysics and quantum computing is needed to achieve the ultimate goal of systems pharmacology-oriented *de novo* drug design for personalized medicine.

1. Introduction

Drug discovery and development is costly and heavily time-consuming process. Besides huge time and money investments, it is infeasible to explore the whole chemical space of drug-like compounds, which is composed by around 10^{33} small molecules [1], by using conventional technologies. Owing to the improvement of computer power, tremendous progresses in deep learning (DL), and emergence of quantum computing, computer-aided drug design has the potential to dramatically speed up the drug discovery process. With a reasonably reliable and accurate computational model, it is possible to synthesis and test a small number of compounds precisely interacting with expected drug target(s) and achieve desirable clinical outcomes. Recently, researchers have introduced various methods to generate novel molecules or optimize existing molecules towards designed properties. The mostly used methods include generative adversarial neural-networks (GAN) [2], variational autoencoder (VAE) [3], normalizing flow [4, 5], and reinforcement learning (RL) [6]. The molecules generated by GAN, VAE, and normalizing flow are biased to specific data distributions. They lack the ability to explore the unknown space that has a distribution shift. RL, on the other hand, is able to tune the pre-trained model specifically toward the properties of interest and enable the model to generate molecules that have different distribution from the training data. However, RL is less efficient compared with other methods. Training a model with RL from scratch will either cost a long time or lead to a model that is hard to converge. Thus, recent studies tend to combine pre-training together with RL to take the advantage of the exploitation ability of transfer learning and the exploration ability of RL.

Existing efforts in applying RL to drug discovery mainly follow conventional one-drug-one-gene target-based paradigm. Although target-based drug discovery is mostly successful in tackling mono-genic diseases whose etiologies are driven by a single gene, it suffers from high failure rate especially for multi-genic, multi-factorial, heterogeneous diseases such as Alzheimer's disease. Moreover, a drug rarely interacts only with its primary target in human bodies. Off-target effects are common, and may contribute to therapeutic effects or side effects [7]. Therefore, systems pharmacology that targets a gene-gene interacting network instead of a single gene and is tailored to individual patients has emerged as a new drug discovery paradigm for complex diseases. However, unlike target-based compound screening that can be easily measured by drug-target binding affinities, advances of systems pharmacology are hindered by the lack of effective read-outs for high-throughput compound screening on an individual basis. Powered by the development of many high-throughput cell-based phenotypic detection methods, phenotype-based drug discovery starts to gain an increasing attention in recent years due to its ability to identify drug lead compounds in a physiologically relevant condition [8]. Phenotype-based drug discovery is a target agnostic and empirical approach to exploit new drugs with no prior knowledge about the drug target or mechanism of action in a disease [9]. The use of molecular signatures as phenotype read-outs makes it possible to not only establish robust drug-disease associations but also deconvolute drug targets from the unbiased phenotype screening. Additionally, phenotype-based drug discovery has the power to exploit drugs for rare or poorly understood diseases such as Alzheimer's disease. Recently, several computational methods have been developed for mechanism-driven phenotype compound screening using chemical-induced gene expressions [10] and images [11] as read-outs. They pave the way of systems pharmacology-oriented high-throughput compound screening for personalized drug discovery.

The review will be organized as follows. We will first give a brief overview of RL, including definitions of some key concepts, problem setting and formulation of leading methods. Then we will survey the recent developments in applying deep RL to drug discovery. Finally, we will highlight the challenges and opportunities of RL in systems pharmacology and personalized medicine. Molecule generation is an important and related topic in *de novo* drug design, but will not be covered in this review. Interested readers can refer to related references [12–19].

2. Overview of reinforcement learning

In reinforcement learning, there are usually two main characters -- an agent and an environment. The agent is the key component of RL that makes sequential decisions, and the environment is the world that the agent lives in. A typical RL problem can be considered as training an agent to interact with an environment that follows a Markov Decision Process (MDP) [20].

In each interaction (with the environment), the agent receives the information of the current state $s_t \in \mathcal{S}$ and performs an action $a_t \in \mathcal{A}$ accordingly, where $\mathcal{S}$ and $\mathcal{A}$ are state and action spaces.[1] After performing an action a_t, the agent will transition to a new state s_{t+1} and receive a reward r_t. These are characterized by underlying state transition dynamics $P: \mathcal{S} \times \mathcal{A} \rightarrow \Delta(\mathcal{S})$ and the reward function $r: \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$, i.e., $P(s_{t+1} | s_t, a_t,)$ and $r(s_t, a_t)$ are the probability and reward of taking action a_t in state s_t and then transitioning into state s_{t+1}. This process repeats indefinitely or until a predefined termination condition is met. The sequence of states and actions followed in this process constitutes a so-called trajectory τ, e.g., at time t, $\tau_t = \{s_1, a_1, s_2, a_2, \ldots, s_t, a_t\}$. Moreover, with a discount factor $\gamma \in (0, 1]$, we can define the discounted cumulative reward under a trajectory τ as $R(\tau) = \sum_{t=1}^{\infty} \gamma^{t-1} r(s_t, a_t)$. An MDP $\mathcal{M}$ can be represented as a tuple of all components mentioned above along with an initial state distribution μ, i.e., $\mathcal{M} = \{\mu, \mathcal{S}, \mathcal{A}, P, r, \gamma\}$.

In the typical setting of MDP, agent behaves by following a (stationary) policy π, which specifies a decision-making strategy in which the agent chooses an action a_t adaptively only based on its current state s_t. Precisely, a stochastic policy is specified as $\pi : \mathcal{S} \rightarrow \Delta(\mathcal{A})$ while a deterministic policy is of the form $\pi : \mathcal{S} \rightarrow \mathcal{A}$.

Given an MDP $\mathcal{M}$ and a policy π, we can define some functions that measures the quality of being in a state s_t or taking an action a_t upon s_t in the long run. Specifically, we can define the value function $V_{\mathcal{M}}^{\pi} : \mathcal{S} \rightarrow \mathbb{R}$ that gives discounted sum of future rewards at a state s_t following an arbitrary policy π:

$$V_{\mathcal{M}}^{\pi}(s) = \mathbb{E}_{\tau \sim T_\pi(\tau)}\left[\sum_{t=1}^{\infty} \gamma^{t-1} r(s_t, a_t) \mid \pi, s_1 = s\right] \quad (1)$$
$$\text{where } T_\pi(\tau) = \mu(s_1) \prod_{t=1}^{\infty} \pi(a_t|s_t) P(s_{t+1}|s_t, a_t)$$

Similarly, the action-value function (or Q-function) $Q_{\mathcal{M}}^{\pi}: \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ can be defined as:

$$Q_{\mathcal{M}}^{\pi}(s, a) = \mathbb{E}_{\tau \sim T_\pi(\tau)}\left[\sum_{t=1}^{\infty} \gamma^{t-1} r(s_t, a_t) \mid \pi, s_1 = s, a_1 = a\right] \quad (2)$$

[1] We use t to track the time-step, e.g., s_t is the state at time *t*

With $V^{\pi}_{\mathcal{M}}(s)$ and $Q^{\pi}_{\mathcal{M}}(s,a)$, we can also define another value function, the advantage function $A^{\pi}_{\mathcal{M}}(s,a) = Q^{\pi}_{\mathcal{M}}(s,a) - V^{\pi}_{\mathcal{M}}(s)$, which measures the relative reward that the agent could obtain by taking a particular step a upon s (compared to an average action).

The objective of RL is to learn a policy π that optimizes the expectation of accumulated reward under a particular initial state distribution μ:

$$\max_{\pi\in\Pi} \mathrm{J}_{\mu}(\pi) = \max_{\pi\in\Pi} \mathbb{E}_{s_1\sim\mu}[V^{\pi}_{\mathcal{M}}(s_1)] \quad (3)$$
$$= \max_{\pi\in\Pi} \mathbb{E}_{\tau\sim T_{\pi}(\tau)}[R(\tau)] \quad (4)$$

Please note in the following text, we drop the subscript of $\mathcal{M}$ on value functions and μ from the objective function $\mathrm{J}_{\mu}(\pi)$ for convenience given working under the same MDP $\mathcal{M}$.

Value-based Methods

One way to optimize the RL objective $\mathrm{J}(\pi)$ is by the observation that if the optimal value function or Q-function could be accurately estimated, we can easily recover an optimal policy. For instance, given the optimal Q-function $Q^*(s,a)$, an optimal policy $\pi^*(s)$ could be obtained by

$$\pi^*(s) = \mathrm{argmax}_{a\in\mathcal{A}} Q^*(s,a) \quad (5)$$

Dynamic Programming is a classic approach to approximate these desired value functions assuming a perfect model of the environment is given, and the number of states and actions is small so that value functions can be represented in some lookup tables [20]. The foundation of DP is centered on bellman optimality equation and bellman expectation equation, and they can be defined as follows with respect to Q-function:

$$Q^*(s,a) = r(s,a) + \gamma\mathbb{E}_{s'\sim P}[\max_{a'} Q^*(s',a')] \quad (6)$$
$$Q^{\pi}(s,a) = r(s,a) + \gamma\mathbb{E}_{a'\sim\pi,s'\sim P}[Q^{\pi}(s',a')] \quad (7)$$

According to these equations, two helpful mathematical operations, bellman optimality operator $\mathcal{T}: \mathbb{R}^{|\mathcal{S}||\mathcal{A}|} \to \mathbb{R}^{|\mathcal{S}||\mathcal{A}|}$ and bellman evaluation operator $\mathcal{T}^{\pi}: \mathbb{R}^{|\mathcal{S}||\mathcal{A}|} \to \mathbb{R}^{|\mathcal{S}||\mathcal{A}|}$ can be constructed as follows:

$$(\mathcal{T}Q)(s,a) := r(s,a) + \gamma\mathbb{E}_{s'\sim P}[\max_{a'} Q(s',a')] \quad (8)$$
$$(\mathcal{T}^{\pi}Q)(s,a) := r(s,a) + \gamma\mathbb{E}_{a'\sim\pi,s'\sim P}[Q(s',a')] \quad (9)$$

These operators map any Q-function from $\mathbb{R}^{|\mathcal{S}||\mathcal{A}|}$ into another Q-function in the same space. It's helpful to consider the Q-function as a vector of length $|\mathcal{S}||\mathcal{A}|$ and the operators are just some transformations that takes the vector and output another vector with same dimensions. More specifically, considering $\mathcal{T}$ and a state-action value $Q(s,a)$, $\mathcal{T}$ can be treated as an assignment statement that replaces the original value of $Q(s,a)$ with the value computed by RHS of equation (8). There are two pleasant properties of these operators, related to contraction mapping, which help the design of the classic DP algorithms. First, given any arbitrary Q-function Q, repeatedly applying $\mathcal{T}^{\pi}$ or $\mathcal{T}$ on Q yields Q^{π} or Q^* respectively. This property can directly turn Bellman operators into update rules, providing an iterative algorithm for approximating a desired Q-function. Moreover, $\mathcal{T}^{\pi}$ and $\mathcal{T}$ have unique fixed points Q^{π} and Q^{*} such that $\mathcal{T}^{\pi}Q^{\pi} = Q^{\pi}$ and $\mathcal{T}Q^{*} = Q^{*}$. This second property serves as the termination condition of many classic DP algorithms and building blocks of some advanced RL algorithms, such as actor-critic.

Since, in practice, the complete knowledge of the environment is usually unknown and the number of states and actions can be arbitrarily large, dynamic programming is thus limited to some restricted problems by its assumptions and function representation. The value-based RL provides a class of algorithms that overcome these problems, combining the framework of iterative dynamic programming with function approximation and extensions of bellman operators. The essence of this approach is to modify the update rule by approximating the bellman operators with some empirical estimators, i.e., estimating the RHS of equations (8) and (9) with sampling [21]. In the case of using a single sample $< s, a, r, s' >$ where $r = r(s,a)$, the empirical bellman operators can be defined as [22]:

$$(\hat{\mathcal{T}}Q)(s,a) := r + \gamma \max_{a'} Q(s',a') \qquad (10)$$
$$(\widehat{\mathcal{T}^{\pi}}Q)(s,a) := r + \gamma Q(s',\pi(s')) \qquad (11)$$

A classic algorithm under this category is the Fitted Q-Iteration [23]. In FQI, the algorithm first gathers a dataset D in the form of tuples $< s, a, r, s' >$ where state s and action a are drawn from a pre-defined distribution ν, and the next state s' and the reward r are obtained from an unknown state transition probability $P(\cdot\,|s,a)$ and reward function $r(s,a)$. With the initialization of a parameterized Q-function Q_{θ}, the Q-values $Q_{\theta}(s,a)$ are updated towards their target $\hat{\mathcal{T}}Q_{\theta}(s,a)$ and the update rule can be formulated as

$$\theta \leftarrow \text{minimize}_{\theta} \sum_{i=1}^{|D|} \left(\hat{\mathcal{T}}Q_{\theta}(s_i,a_i) - Q_{\theta}(s_i,a_i)\right)^2 \qquad (12)$$
Where $\hat{\mathcal{T}}Q_{\theta}(s_i,a_i) = r + \gamma \max_{a'} Q_{\theta}(s_i', a_i')$

Equation (12) can be interpreted as applying Monte-Carlo approximation of Bellman optimality $\hat{\mathcal{T}}$ to Q-function Q_{θ} through minimizing the square loss over θ. And since the Q-values in FQI are estimated by a parameterized function rather than a lookup table (as in DP), they are closely correlated to each other, implying that a small update of θ may benefit some Q-values but push others away from their targets. Therefore, unlike supervised learning where the ground truth labels remain unchanged, the targets $\hat{\mathcal{T}}Q$ defined in FQI may vary every time when the parameter changes, which introduces instability in training. Consequently, depends on the generalization and extrapolation ability of the function approximation, the contraction mapping property of DP cannot guarantee convergence under parametrized Q-function, especially for those using neural networks [24]. However, there exist other variants of FQI using non-parametric approximation architecture, such as k-nearest-neighbor and totally randomized trees, showing strong convergence property [25–28].

Regarding to the value-based methods using neural networks, Deep Q-Network algorithm, showing strong performance in a variety of ATARI games, can be viewed as an instantiation of FQI in online setting [29]. The Q-function approximator of DQN can be any typical neural network, referred to as Q-network. The framework of DQN uses two heuristics to limit the instability inherited from FQI with neural networks. First, a separated network called target network is introduced solely for computing the targets (interpreted as the ground truth) due to their inconsistency. Compared to Q-network, target network is updated less frequently for keeping the target fixed for extra amount of time. With this strategy, DQN prevents the instability to prorogate too quickly and thus reduces the risk of divergence.

Moreover, Deep Q-Network is an online learning algorithm which follows its current policy for exploring and data sampling. Without proper care, this may deliver a policy that has inferior performance since previously visited state-action pairs with high rewards may not guarantee to be revisited due to the stochasticity of the system. Experience Replay, a replay buffer, is introduced to solve this issue by keeping all samples $< s, a, r, s' >$ from the last N steps in the memory where N is usually very large. Besides, ER selects samples from the memory following the uniform distribution for every mini-batch update. This sampling method, though simple, effectively breaks the temporal correlation between samples in the same trajectory, which helps reduce bias. With large memory and uniformly sampling, mini-batch samples constructed under ER are more representative than that of alternative methods such as Q-learning [30] which uses only one sample for an update. Additionally, large buffer provides good coverage of the state-action space, which makes the policy training more exploratory and consistent, thus increasing the chance of achieving a more desirable policy.

In addition to target network and experience replay, there are other advanced approaches that could help improve the stability and efficiency of learning including DDQN [31] for overestimation reduction, Multi-step learning for accurate target estimation [24] and PER for efficient TD-error minimization [32], along with other extensions such as Distributional DQN [33] and Dueling Networks [34].

Policy gradient

Unlike the value-based approach that learns a desired policy by estimating the optimal value functions, the Policy Gradient methods work directly on the objective function defined in (4) or some equivalent ones. In the setting of this review, we limit our focus on (4). Specifically, with a policy π parameterized by $\theta \in \Theta \subset \mathbb{R}^d$, the goal of PG is to find a θ that maximizes $J(\pi_\theta)$:

$$\max_{\theta \in \Theta} J(\pi_\theta) \tag{13}$$

Since the search space has now shifted from the policy space Π in (4) to Θ which is continuous and has fixed dimensions, various numerical optimization methods could be utilized to solve (13). Gradient ascent is one direct approach to this, which iteratively move in the direction of steepest ascent as defined by the gradient for maximizing $J(\pi_\theta)$. Following is the expression of the gradient for $J(\pi_\theta)$:

$$\begin{aligned} \nabla_\theta J(\pi_\theta) &= \nabla_\theta \mathbb{E}_{\tau \sim T_{\pi_\theta}(\tau)}[R(\tau)] \\ &= \mathbb{E}_{\tau \sim T_{\pi_\theta}(\tau)}[\textstyle\sum_{t=1}^{\infty} \nabla_\theta \log \pi_\theta(a_t|s_t) R(\tau)] \end{aligned} \tag{14}$$

Where $R(\tau) = \sum_{t=1}^{\infty} \gamma^{t-1} r(s_t, a_t)$.

At each update iteration k, gradient ascent, with a fixed step size α, follows the update rule:

$$\theta_{k+1} = \theta_k + \alpha \nabla_{\theta_k} \mathrm{J}(\pi_{\theta_k}) \tag{15}$$

Since the policy gradient $\nabla_\theta \mathrm{J}(\pi_\theta)$ is an expectation over all possible trajectories, it can be directly estimated by a set of samples where each is a trajectory collected by the agent interacting with the environment while following policy π_θ. With the set of samples denoted as D and the length of trajectories assumed to be H, we can estimate $\nabla_\theta \mathrm{J}(\theta)$ as:

$$\nabla_\theta \mathrm{J}(\theta) \approx \frac{1}{|D|} \sum_{\tau \in |D|} \nabla_\theta \log T_{\pi\theta}(\tau)[R(\tau)] \tag{16}$$

$$\approx \frac{1}{|D|} \sum_{i=1}^{|D|} \sum_{t=1}^{H} \nabla_\theta \log \pi_\theta\left(a_{i,t} \middle| s_{i,t}\right) R(\tau) \tag{17}$$

Where $R(\tau) = \sum_{t=1}^{H} \gamma^{t-1} r(s_t, a_t)$.

With equation (17), we have derived our first policy gradient method, commonly known as REINFORCE [35] which serves as the foundation of PG-based methods.

Because $\nabla_\theta \mathrm{J}(\pi_\theta)$ is an expectation, a longstanding issue centered on PG-based methods is to derive an unbiased estimator of $\nabla_\theta \mathrm{J}(\pi_\theta)$ with possibly low variance. For instance, $R(\tau)$ in (17) can be replaced by a smaller term $\sum_{t'=t}^{H} \gamma^{t'-t} r(s_{t'}, a_{t'})$ thanks to causality that policy at time t' cannot affect reward at time t when $t < t'$. This gives us a new unbiased estimator of $\nabla_\theta \mathrm{J}(\pi_\theta)$ with reduced variance called Reward-to-go [36]:

$$\nabla_\theta \mathrm{J}(\theta) \approx \frac{1}{|D|} \sum_{i=1}^{|D|} \sum_{t=1}^{H} \gamma^{t-1} \nabla_\theta \log \pi_\theta\left(a_{i,t} \middle| s_{i,t}\right) \left(\sum_{t'=t}^{H} \gamma^{t'-t} r(s_{i,t'}, a_{i,t'})\right) \tag{18}$$

Regarding the policy gradient $\nabla_\theta \mathrm{J}(\pi_\theta)$, it can be viewed as an analogue of the gradient in maximum likelihood where the term $\log \pi_\theta\left(a_{i,t} \middle| s_{i,t}\right)$ in equations (17) and (18) is the log likelihood of the policy π_θ given data $<s_{i,t}, a_{i,t}>$. Unlike maximum likelihood, the log likelihood in PG is now weighted by some discounted cumulative rewards, such as $R(\tau)$ or $\sum_{t'=t}^{H} \gamma^{t'-t} r(s_{i,t'}, a_{i,t'})$. Thus, updating the parameter θ with $\nabla_\theta \mathrm{J}(\pi_\theta)$ would change the policy according to the sign of $R(\tau)$ or $\sum_{t'=t}^{H} \gamma^{t'-t} r(s_{i,t'}, a_{i,t'})$, e.g., the chance of selecting $a_{i,t}$ given $s_{i,t}$ increases if $R(\tau)$ is positive or decreases otherwise.

On the other hand, if the reward function only outputs positive values and some mediocre actions are randomly selected for parameter updates, the agent may tend to choose these inferior actions even their absolute values of $\log \pi_\theta\left(a_{i,t} \middle| s_{i,t}\right)$ are small. A popular strategy to address this issue is to introduce an extra term, called baseline *b*, which can be shown to always keep the estimator of $\nabla_\theta \mathrm{J}(\theta)$ unbiased while reduce its variance if selected properly [37]. For instance, by choosing an average reward over sampled trajectories as baseline, the term $[\left(\sum_{t'=t}^{H} \gamma^{t'-t} r(s_{i,t'}, a_{i,t'})\right) - b]$ in (19) is centered around 0, which improves the sampling efficiency by distinguishing actions with their relative rewards.

$$\nabla_\theta \mathrm{J}(\theta) \approx \frac{1}{|D|} \sum_{i=1}^{|D|} \sum_{t=1}^{H} \gamma^{t-1} \nabla_\theta \log \pi_\theta\left(a_{i,t} \middle| s_{i,t}\right) [\left(\sum_{t'=t}^{H} \gamma^{t'-t} r(s_{i,t'}, a_{i,t'})\right) - b] \tag{19}$$

Where *b* can be $\frac{1}{|D|} \sum_{i=1}^{|D|} R(\tau)$

Thus, with proper care of designing the baseline, the policy π_θ could eliminate the inferior actions and choose more desirable ones with higher probability. In the section of actor-critic, a learnable baseline will be introduced, which is more stable and significantly reduces the variance of policy gradient estimators.

Actor-Critic

Actor-critic algorithms can be viewed as an approach that combines policy gradients with value-based methods to optimize $\mathrm{J}(\theta)$. Generally, it takes a policy π_θ as an actor to interact with the environment, while maintain a learnable value function as the critic to evaluate the actor's actions [24, 38]. The simplest form of actor-critic called Q actor-critic [38] can be derived directly from the reward-to-go and we denote $\widehat{\nabla_\theta \mathrm{J}(\theta)}$ below as the estimator of $\nabla_\theta \mathrm{J}(\theta)$ from (18):

$$\begin{aligned}
\mathbb{E}\left[\widehat{\nabla_\theta \mathrm{J}(\theta)}\right] &= \mathbb{E}\left[\frac{1}{|D|}\sum_{i=1}^{|D|}\sum_{t=1}^{H}\gamma^{t-1}\widehat{Q^{\pi_\theta}}(s_t,a_t)\nabla_\theta \log \pi_\theta\left(a_{i,t}\middle|s_{i,t}\right)\right] \\
&= \mathbb{E}\left[\frac{1}{|D|}\sum_{i=1}^{|D|}\sum_{t=1}^{H}\gamma^{t-1}\mathbb{E}\left[\widehat{Q^{\pi_\theta}}(s_t,a_t)\middle|s_t,a_t\right]\nabla_\theta \log \pi_\theta\left(a_{i,t}\middle|s_{i,t}\right)\right] \\
&= \mathbb{E}\left[\frac{1}{|D|}\sum_{i=1}^{|D|}\sum_{t=1}^{H}\gamma^{t-1}Q^{\pi_\theta}(s_t,a_t)\nabla_\theta \log \pi_\theta\left(a_{i,t}\middle|s_{i,t}\right)\right] \quad (20)
\end{aligned}$$

Where $\widehat{Q^{\pi_\theta}}(s_t,a_t) \; = \; \sum_{t'=t}^{H}\gamma^{t'-t}r(s_{i,t'},a_{i,t'})$

A direct advantage of this new estimator, the one inside the expectation of (20), is that it has less variance than the reward-to-go. This can be shown following tower property of conditional probability and the definition of variance. The intuition is that $\widehat{Q^{\pi_\theta}}(s_t,a_t)$ is indeed the cumulative reward of a single sample trajectory that estimates $Q^{\pi_\theta}(s_t,a_t)$, which inevitably has larger variance and so as the reward-to-go. In practice, we usually estimate Q^{π_θ} by a parameterized Q-function $Q_\omega^{\pi_\theta}$ and update ω following a similar procedure from Fitted Q-Iteration described in the section of value-based methods.

Like (19), a baseline can be incorporated into Q actor-critic to further reduce its variance. A favorable choice is to use a value function $V_\varphi^{\pi_\theta}$ parameterized by φ where $\nabla_\theta \mathrm{J}(\pi_\theta)$ can be further reduced to a form of advantage function:

$$\begin{aligned}
\nabla_\theta \mathrm{J}(\theta) &\approx \tfrac{1}{|D|}\textstyle\sum_{i=1}^{|D|}\sum_{t=1}^{H}\gamma^{t-1}\nabla_\theta \log \pi_\theta\left(a_{i,t}\middle|s_{i,t}\right)\left[Q_\omega^{\pi_\theta}\left(s_{i,t},a_{i,t}\right) - V_\varphi^{\pi_\theta}\left(s_{i,t}\right)\right] \quad (21) \\
&\approx \tfrac{1}{|D|}\textstyle\sum_{i=1}^{|D|}\sum_{t=1}^{H}\gamma^{t-1}\nabla_\theta \log \pi_\theta\left(a_{i,t}\middle|s_{i,t}\right) A_{\omega,\varphi}^{\pi_\theta}\left(s_{i,t},a_{i,t}\right)
\end{aligned}$$

The derived PG estimator in (21) is known as the advantage actor-critic [24, 39, 40]. As the advantage function measures the performance difference between a specific action and the average action in a given state, it's considered as a favorable critic to evaluate the actor and determine which action should be chosen more often.

Moreover, $\nabla_\theta J(\pi_\theta)$ can be approximated in another way as specified in (22) by the observation that given a sample $< s, a, r, s' >$, the sum of r and $\gamma V^{\pi}(s')$ is a sample estimate of $Q^{\pi}(s, a)$:

$$\nabla_\theta J(\theta) \approx \frac{1}{|D|} \sum_{i=1}^{|D|} \sum_{t=1}^{H} \gamma^{t-1} \nabla_\theta \log \pi_\theta(a_{i,t} | s_{i,t}) \left(r(s_{i,t}, a_{i,t}) + \gamma V_{\varphi}^{\pi_\theta}(s_{i,t+1}) - V_{\varphi}^{\pi_\theta}(s_{i,t}) \right) \quad (22)$$

The unbiased estimator of $\nabla_\theta J(\pi_\theta)$ derived in (22) is called TD-error actor-critic [41]. Theoretically, it has a higher variance than advantage actor-critic as we replace $Q_{\omega}^{\pi}(s_{i,t}, a_{i,t})$ with a single sample estimate $r(s_{i,t}, a_{i,t}) + \gamma V_{\varphi}^{\pi_\theta}(s_{i,t+1})$. But in practice the estimator of $\nabla_\theta J(\pi_\theta)$ in (22) is more stable for policy training and has less bias as it only requires a function estimator for the value function $V_{\varphi}^{\pi_\theta}$ while advantage actor-critic requires another function estimator $Q_{\omega}^{\pi_\theta}$ for Q-function other than $V_{\varphi}^{\pi_\theta}$, which thus needs more samples to achieve a comparable performance.

3. State-of-the-arts in applying reinforcement learning to drug discovery

To generate molecules distributing far away from training data, researchers start to use RL together with DL models for molecule generation and optimization. The basic idea is to get a DL model with a latent space that has the same distribution with the training dataset with pre-training. Then use RL to introduce bias skewing the distribution close to desire properties, which can be totally different than the training data. RL architectures used in drug discovery majorly fall into three classes, value-based deep Q-networks, policy-based policy gradient methods, and a mix of value- and policy-based actor-critic method. Value-based RL usually is more efficient and stable to train. However, policy gradient is adaptive to a wider range of problems, especially problems with large action space. Actor-critic improve the sampling efficiency of policy gradient by learning a parameterized value function (critic).

Models with policy gradient

For molecule generation, because of the large searching space, policy gradient based RL become the most widely used architecture. Marcus et al. develop a policy-gradient-based approach in which they pre-train a prior RNN with ChEMBL dataset to generate SMILES with the same structure distribution as ChEMBL [42]. Another agent RNN copies the architecture and parameters from the pre-trained prior RNN is then tuned with RL to achieve higher expected score while keep the new policy to be anchored to the prior's policy so that generated novel molecules have the same structure distribution as ChEMBL. Since parameters of RNN are pre-trained, the searching efficiency of RL is increased. However, for the same reason, the exploration capability of this model is limited.

Recently, polypharmacology attracts more and more researcher's attention because its potential therapeutic ability in some complex pathologies. Dual-target ligand generative network (DLGN) is developed to generate molecules that have bioactivities toward two targets [43]. With SMILES as input, DLGN uses an RNN-based generator to produce novel molecules that also in

SMILES format. To make generated molecules dual-targeting, DLGN utilizes two discriminators to monitor the activities of newly generated molecules toward two targets, respectively. The shortcoming of this model is, although the generator does not need to be trained with labeled data, the discriminators require high quality labeled data to control qualities of generated molecules.

As a more natural way to represent molecules, graphs, thus GNNs, are widely used in machine-based drug discovery. Graph convolutional policy network (GCPN) generates molecules under the guidance towards desired objectives, while restricting the output space to specified chemical rules [44]. To achieve the goal, they use RL to represent constraints and desired properties through the design of environment and reward function, and use adversarial training to incorporate prior knowledge of specified dataset. Another interesting work that utilizes graph data structure to represent molecules while exploiting RNN as generator is MolecularRNN, in which they use two RNNs termed graph-level RNN and edge-level RNN to generate next atom and bond based on atoms and bonds that are already generated [45]. Their model is also tuned with policy-gradient-based RL. With valency-based rejection sampling constraints, their model yields 100 % validity.

Most molecule generation algorithms do not take 3D structure, which is important in determining drug-target interaction, into consideration. MOLGYM generates molecules in Cartesian coordinates [46]. This enables more accurate molecule physical property prediction such as electronic energy with fast quantum-chemical methods.
Generating molecules in silicon is far from the final goal of developing drugs. The first step after molecular design is to synthesis designed compounds and test their effect with wet experiments. However, not all computational designed molecules are synthesizable. This seriously affects the practical value of computational drug design. Some studies try to resolve this by taking synthesizability into consideration while generating molecules. Reaction-driven objective reinforcement (REACTOR), for example, define the environment's state transitions as sequences of chemical reactions in RL process, and thus not only improve the synthesizability of generated molecules, but also speed up the exploration rate of the model in chemical space [46].

Policy-gradient-based RL has disadvantages of low efficiency and unstable, especially for large searching space as chemical space. In the work REINVENT, Guo et al. uses curriculum learning in molecule generation to increase searching efficiency by setting a series of task with increasing complexity [47]. In addition, comparing to standard policy based RL, REINVENT is capable to take advantage of complex reward functions to generate molecules satisfying multiple constraints.

Models with Deep Q-networks (DQNs)

RL based on value function learning is usually more stable and sample efficient than methods that use policy gradient [29]. Molecule Deep Q-Networks (MolDQN) formulate molecule generation as a Markov Decision Process (MDP) and solve the problem with DQN [48]. By allowing only valid actions, their model promises the generated molecules are 100 % valid. The molecule generation/optimization starts from a molecule base input to neural networks as Morgan fingerprint. With constrains, the output molecule will have one property improved comparing to the original molecule while maintaining similarity above a given level. The advantage of MolDQN is that it does not depend on a pre-trained model for molecule generation, thus is not biased to observed chemical space. Thus, MolDNQ can in principle generate novel chemical structures.

Tang et al. develop an advanced deep Q-learning network with the fragment-based drug design (ADQN-FBDD) to specifically generate molecules targeting a protein with known 3D structure [49]. They design a practical reward based on drug-likeness, containing specific fragments, and pharmacophore, among which specific fragments and pharmacophore are protein structure dependent. Although this approach has very obvious limitation, which is the high dependency of 3D structures of target proteins, recent great progress in protein structure prediction may give it a wide range of application scenarios [50].

Models with actor-critic

Actor-critic method uses two networks. Actor decides what action to take based on the policy, current state and the environment. Critic evaluate the action taken by the actor and inform the actor how it should be adjusted. In most cases, the actor network is also trained with policy gradient. With the guidance from critic, the training process usually becomes more stable and efficient [51]. Reinforcement Learning for Structural Evolution (ReLeaSE) is an example of the usage of actor-critic in drug discovery [52]. ReLeaSE uses two models termed generative model and prediction model. The generative model is a Stack-RNN pretrained with millions of SMILES strings and the prediction model is an RNN with LSTM layers pretrained with desired molecular properties. After pretraining, the generative model is fine-tuned with the prediction model as critic.

Multi-constraint molecular generation (MCMG) (Jike Wang, 2021) uses transformer architecture to read and generate SMILES strings. After pre-training, to improve the search efficiency and stability of RL process, they distilled the transformer with a smaller RNN with the distilled likelihood or the distilled molecules approach. The distilled model is then fine-tune with RL with a property prediction model works as critic [53].

Remarks on RL algorithms

With the jumbo chemical searching space and limited pretraining samples, machine learning algorithms like GAN and VAE have difficulties to generate molecules that evenly distribute in the whole chemical space. Under such a circumstance, RL, which does not rely on training data, can be used to introduce bias to the model and skew the generated molecules to a distribution that is specified in the rewarding function. However, value-based RL lacks the ability to search the whole chemical space while policy-gradient-based RL is inefficient in searching large action space, especially under multiple constraints, which usually is the case for drug-like molecules. To overcome these difficulties, recent studies usually use the pre-training + RL strategy. The neural networks are pre-trained with a training dataset first to get a prior distribution of drug-like molecules, and then RL introduces bias based so that the tuned model can generate molecules with desired properties. The quality of the molecules generated by these strategies highly rely on the quality of the rewarding function. Thus, in future studies, improving the accuracy of property predicting model and searching efficiency of RL process will be the key to bring computational drug design to practical production.

4. Challenges and opportunities of reinforcement learning in systems pharmacology and personalized medicine

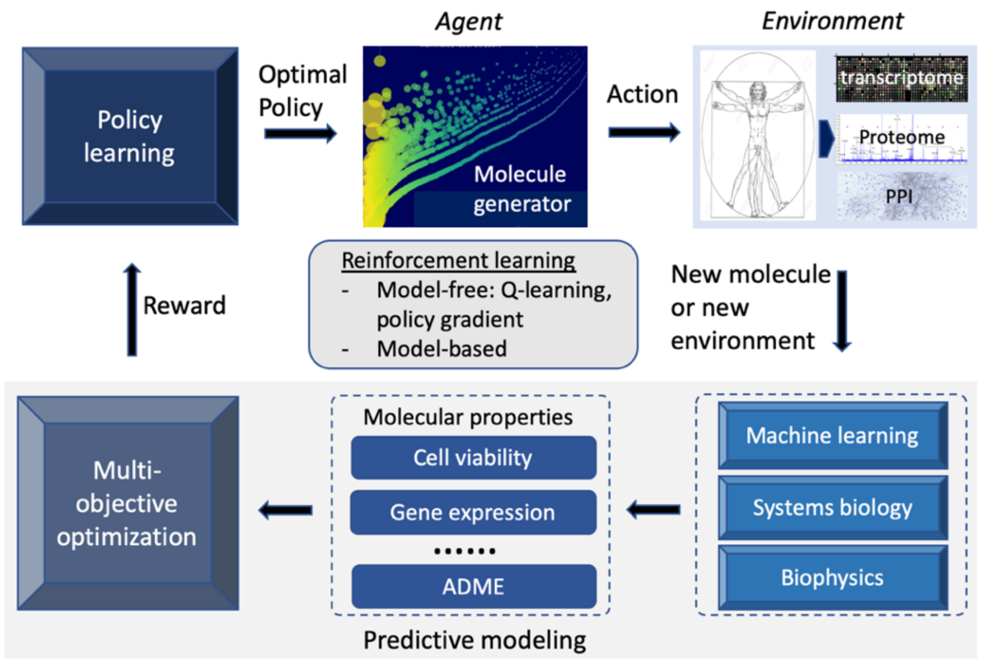

Figure 1. Illustration of a RL framework for systems pharmacology-oriented lead optimization.

Figure 1 illustrates a RL framework for systems pharmacology-oriented personalized lead optimization and drug design. A molecule generator (agent) will first take an action to generate a new molecule or modify a starting molecule (e.g., replacing a hydrogen atom with a methyl group) based on a policy. Then, multiplex phenotypic responses (cell viability, drug-target profile, chemical-induced gene expression, pharmacokinetics etc.) in an individual patient (environment) will be predicted for the newly generated molecule by machine learning, biophysics, systems biology, or other methods. A multi-objective Pareto optimization is performed to determine a reward between state transitions. The molecule and the environment together consist of a state. A new policy will be learned based on observed actions, states, and rewards by a RL algorithm, and a new molecule will be generated again from the updated policy. Unlike target-based compound screening where only chemical space is needed to be explored, systems pharmacology-oriented personalized drug discovery needs to optimize the interplay of chemicals, the druggable genome, and high-dimensional omics characterizations of disease models or patients (environment). Several barriers need to be overcame when applying RL to systems pharmacology and precision medicine. They include the exploration of out-of-distribution samples, generalization power of RL, adaptive multi-objective optimization, , and activity cliffs of quantitative structure-activity relationship (QSAR) space.

Out-of-distribution reward function. Although rewards for several molecular properties (e.g., logP and druglikeness) can be directly calculated from a given molecular structure, reward functions that are the most important for drug actions such as binding affinity and chemical-induced gene expression need to be obtained from a predictive model that is dependent on machine learning, mathematical or physics-based modeling. In spite of tremendous advances in deep learning and availability of diverse omics data sets, robust and accurate predictions of genome-wide drug-target interactions and molecular phenotypic read-outs in a physiological relevant condition remain an unsolved challenging problem. The problem is rooted in biased, noisy, and incomplete omics data and inherent limitation of machine learning. In the case of genome-wide drug-target prediction, only less than 10% of gene families have known small molecule ligands. The remaining 90% gene families are dark matters in the chemical genomics space [54]. Even for mostly studied G-protein coupled receptors (GPCRs), more than 99% receptors are orphans, i.e., their endogenous or exogenous ligands are unknown. Similarly, only a small number of cell lines have annotated drug response data. It is a fundamental challenge to generalize a ``well-trained" machine learning model to unseen data (e.g., patients) that lies out-of-the-distribution (OOD) of the training data (e.g., cell lines), in order to successfully predict outcomes from conditions that the model has never before encountered. While deep learning is capable, in theory, of simulating any functional mapping, its generalization power is notoriously limited in the case of distribution shifts [55].

Generalization power of RL. Markov decision process (MDP) that is the foundation of existing RL algorithms is deterministic and memory-less. It works well in a dense-reward and fully determined environment observed from training data, but will fail catastrophically if the environment is sparse-reward or partially observed [56] or changed (e.g., a new drug target or a new patient). In a dense-reward environment, every action corresponds to a specific reward. On a contrary, if a specific reward is not available, the environment is sparse-reward. The sparse-reward and partially observed or changing environment is the exact situation in systems pharmacology. Due to observed chemical activity and omics data highly biased and incomplete, it is likely that a novel molecule is an OOD sample. Thus, no reliable reward can be assigned to this molecule as mentioned above. Additionally, drug response data mainly comes from cell lines or disease models. The environment in a cell line or animal model could be dramatically different from human bodies. A naïve adaption of standard RL systems is not sufficient for systems pharmacology. New methods are needed to improve the robustness, generalizability and transferability of RL.

Adaptive multi-objective optimization. To design drugs that optimize the system-level responses to maximize therapeutic effects and minimize side effects, it is needed for a RL algorithm to optimize multiple (sometimes conflict) objectives such as pharmacokinetics, blood-brain barrier permeation, drug binding affinities to multiple targets, or chemical-induced gene expression profiles. Although RL methods have been developed for multiple objectives [57], the final reward function in these methods is a linear combination of the reward from each individual objective with *a priori* defined weights. It is often difficult to define such weights. Moreover, the weight is changed when an environment is alternated. For example, gene expression profiles can be dramatically different for different patients and in different disease states. A RL model needs to be trained for different conditions if the weight is fixed. Multi-objective RL in the framework of Pareto optimization is required for systems pharmacology and precision medicine.

Activity cliff of QSAR. Existing RL models for the molecular design are not robust to dramatic environmental changes. In other words, it assumes that the landscape of QSAR is smooth. However, this assumption does not hold in the drug design. A well-known phenomenon in QSAR is activity cliff, in which a slight modification of chemical structure may lead to a dramatic activity change. The activity cliff is more complicated in systems pharmacology than single-targeted drug discovery. For example, the replace of a methyl group with an ethyl group for a chemical compound that have moderate binding affinities to two targets may increase the binding affinity to one of targets, but completely destroy the binding to another target due to steric clash. As a result, the phenotype response modulated by this compound could be changed significantly.

To address aforementioned challenges in adapting RL to systems pharmacology and precision medicine, a synergistic integration of latest advances in RL with new development in machine learning in general and other related fields is needed.

Improving robustness, generalizability, and transferability of RL algorithms. Recently, Ghosh et al, showed that optimal generalization of RL at test-time corresponds to solving a partially-observed Markov decision process (POMDP) that is induced by the agent's epistemic uncertainty about the test environment. They proposed an algorithm, LEEP, which uses an ensemble of policies to approximately learn the Bayes-optimal policy for maximizing test-time performance. In another study, Agarwal et al. proposed meta-reward learning to achieve the generalizability of RL in a sparse-reward environment [58]. In principle, these techniques can be adapted for systems pharmacology.

The majority of existing work in RL for drug design is based model-free approach. Model-based RL [6] may provide new opportunities on addressing challenges in systems pharmacology-oriented drug discovery for precision medicine. Different from model-free approach, model-based methods use optimization search to improve policy by estimating transition probability $p(s'|s, a)$ and optimizing $\pi_\theta(a|s)$. It is more powerful in predicting future reward, has higher sample efficiency, and bears stronger transferability & generality than the model-free approach. The ability of predicting future reward may facilitate avoiding activity cliffs. Sample efficiency will be helpful to alleviate issues in data sparsity, biasness, and noisiness. Transferability and generality are critical to address the OOD problem.

Several algorithms have been developed to solve multi-objective optimization problem in RL. Yang et al. introduced an envelope Q-learning method that can quickly adapt and solve new tasks with different preferences [59]. The capability of learning a single Q function over all preferences is important for personalized drug discovery. In another study, Chen et al. proposed a two-stage model [60] for multi-objective deep RL. At the first stage, a multi-policy soft actor-critic algorithm is applied to collaboratively learn multiple policies with different targets, in which each policy targets on a specific scalarized objective. At the second stage, a multi-objective covariance matrix adaptation evolution strategy is applied to fine-tune the policy-independent parameters.

Improving reward functions for OOD data. A plethora of machine learning approaches including self-supervised learning, transfer learning, semi-supervised learning, meta-learning and their combinations have been recently developed to address out-of-distribution problems in compound screening in terms of chemicals, proteins, and cell lines. Self-supervised learning has enjoyed a great success in Natural Language Processing and image recognition, and adapted to model protein sequences. Cai et al. has proposed a DIstilled Sequence Alignment Embedding (DISAE) transformer for predicting ligand binding to orphan receptors [61]. DISAE has been further extended to out-of-distribution prediction of receptor activities of ligand binding, specifically, agonist vs antagonist [62]. An out-of-cluster meta-learning algorithm has been proposed to explore dark chemical genomics space that includes all Pfam families [63]. Self-supervised learning and semi-supervised learning have also been applied to explore chemical space [64]. However, the performance improvement in QSAR is moderate partly due to the activity cliff mentioned above. Transfer learning is particularly useful in predicting drug responses (both cell viability and gene expressions) for novel cell lines [65] or translate *in vitro* compound screens to clinical outcomes in patients [66]. These methods, when applied to reward functions, could improve the performance of RL in systems pharmacology.

Biophysics-based methods such as molecular dynamics (MD) simulation, quantum chemistry, and protein-ligand docking (PLD), can be directly applied to evaluate the chemical properties, thus used as reward functions. MD simulation and quantum chemistry calculation are computationally expensive. Emergence of quantum computing may make it feasible to incorporate them directly into RL as reward functions [67]. With the advent of high-accuracy protein structural models, predicted by AlphaFold2 [50], it now becomes feasible to use PLD to predict ligand-binding sites and poses on dark proteins, on a genome-wide scale. However, PLD suffers from a high false-positive rate due to poor modeling of protein dynamics, solvation effects, crystallized waters, and other challenges [68]; often, small-molecule ligands will indiscriminately `stick' to concave, pocket-like patches on protein surfaces. For these reasons, although AlphaFold2 can accurately predict many protein structures, the relatively low reliability of PLD still poses a significant limitation, even with a limitless supply of predicted structures [69]. Thus, the direct application of PLD remains a challenge for predicting ligand binding to dark proteins. Recently, Cai et al. have proposed an end-to-end sequence-structure-function learning framework PortalCG [54]: protein structure information is not used as a fixed input, but rather as an intermediate layer that can be tuned using various structural and functional information. PortalCG significantly outperforms the direct use of PLD for predicting ligand binding to dark proteins. Thus, it could be an effective strategy to incorporate biophysics domain-knowledge into deep learning framework as constraints or regularizations.

Conclusion

In spite of the great success of RL in GO, computer games, and other settings with well-defined environments, its application to drug discovery is still in its infancy. Current efforts in RL mainly focus on target-based drug design. Although RL is a promising technique, its actual value in the target-based drug discovery is over-optimized in the current stage. The major hurdle comes from the reward function that is based on predicted binding affinities and needs to be obtained from either a machine learning approach or a physics-based scoring. Both of them are not reliable and accurate enough when applied to chemical compounds with novel structures. For a machine learning approach, it remains a great challenge to predict an OOD case, i.e., a generated molecule that falls outside the distribution of chemicals in the training data for activity predictions. As a result, the structure of active compound inferred from RL may be not significantly different from those in the training data. In terms of physical-based scoring, there is lack of computationally efficient and accurate methods to model multiple factors including conformational dynamics, solvent effects, hydrogen bonding, entropies, crystallized waters, etc., which contribute to the protein-ligand binding affinity. Consequently, the scoring function is still suboptimal and unreliable. In authors' humble opinion, existing RL approaches to *de novo* drug design is scarcely fruitful in regard to the structural novelty of chemical compounds without the significant improvement of efficiency and accuracy of binding affinity predictions.

Although current efforts in RL mainly center around the target-based drug design, systems pharmacology and personalized medicine have emerged as new paradigms in drug discovery. They have advantages over conventional "one-drug-one-gene" approach when tracking multi-genic, heterogenous diseases. Systems pharmacology-oriented and personalized drug design imposes new challenges on RL. Conceptually, systems pharmacology has not been fully appreciated by pharmaceutical and biotechnology industry. Technically, there are few high-quality labeled data available for training a generalizable machine learning model to predict molecular phenotypic readouts suitable for systems pharmacology-oriented and personalized drug design. Thus, the OOD problem in systems pharmacology is more serious than the target-based drug design. Besides the OOD issue that incapacitates the reward function, the success of RL in systems pharmacology-oriented and personalized drug design needs to overcome additional roadblocks. Firstly, the generalizability and transferability of RL itself will be a serious problem, because RL environments are partially observable or changed dramatically (e.g., from cell lines to human tissues) in the setting of systems pharmacology. Secondly, RL needs to optimize multiple dynamic and often conflict rewards in a dynamic environment. Multi-objective dynamic reward function optimization remains an unsolved research problem. Finally, it is necessary to integrate multiple heterogeneous, noisy, and high-dimensional omics data for successful systems pharmacology modeling. It is a challenging task under intensive investigations. Thus, the realization of the full potential of RL in drug discovery relies on not only new developments in RL but also advances in other fields that are beyond RL. RL needs to be synergistically integrated with unsupervised/supervised machine learning, biophysics, systems biology, multi-omics technology, and quantum computing.

References

1. Polishchuk PG, Madzhidov TI, Varnek A. Estimation of the Size of Drug-like Chemical Space Based on GDB-17 Data. *J. Comput. Aided. Mol. Des.* 2013 August; *27* (8):675–679.
2. Goodfellow I, Pouget-Abadie J, Mirza M, et al. Generative Adversarial Networks. *Commun. ACM* 2014 June 10; *63* (11):139–144.
3. Kingma DP, Welling M. Auto-Encoding Variational Bayes. *2nd Int. Conf. Learn. Represent. ICLR 2014 - Conf. Track Proc.* 2013 December 20;
4. Rezende DJ, Mohamed S. Variational Inference with Normalizing Flows. *32nd Int. Conf. Mach. Learn. ICML 2015* 2015 May 21; *2*:1530–1538.
5. Madhawa K, Ishiguro K, Nakago K, et al. GraphNVP: An Invertible Flow Model for Generating Molecular Graphs. 2019 May 28;
6. Pack Kaelbling L, Littman ML, Moore AW, et al. Reinforcement Learning: A Survey. *J. Artiicial Intell. Res.* 1996; *4*:237–285.
7. Xie L, Xie L, Kinnings SL, et al. Novel Computational Approaches to Polypharmacology as a Means to Define Responses to Individual Drugs. *Annu. Rev. Pharmacol. Toxicol.* 2012; *52*:361–379.
8. Swinney DC, Lee JA. Recent Advances in Phenotypic Drug Discovery. *F1000Research* 2020; *9*:.
9. Moffat JG, Vincent F, Lee JA, et al. Opportunities and Challenges in Phenotypic Drug Discovery: An Industry Perspective. *Nat. Rev. Drug Discov. 2017 168* 2017 July 7; *16* (8):531–543.
10. Pham TH, Qiu Y, Zeng J, et al. A Deep Learning Framework for High-Throughput Mechanism-Driven Phenotype Compound Screening and Its Application to COVID-19 Drug Repurposing. *Nat. Mach. Intell.* 2021 March 1; *3* (3):247–257.

11. Méndez-Lucio O, Zapata PAM, Wichard J, et al. Cell Morphology-Guided De Novo Hit Design by Conditioning Generative Adversarial Networks on Phenotypic Image Features. 2020 January 17;
12. Mauri A. AlvaDesc: A Tool to Calculate and Analyze Molecular Descriptors and Fingerprints. *Methods Pharmacol. Toxicol.* 2020;801–820.
13. Moriwaki H, Tian YS, Kawashita N, et al. Mordred: A Molecular Descriptor Calculator. *J. Cheminform.* 2018 February 6; *10* (1):1–14.
14. Rogers D, Hahn M. Extended-Connectivity Fingerprints.
15. Duvenaud D, Maclaurin D, Aguilera-Iparraguirre J, et al. Convolutional Networks on Graphs for Learning Molecular Fingerprints. *Adv. Neural Inf. Process. Syst.* 2015 September 30; *2015-Janua*:2224–2232.
16. Guimaraes G, Sanchez-Lengeling B, Outeiral C, et al. Objective-Reinforced Generative Adversarial Networks (ORGAN) for Sequence Generation Models. 2017 May 30;
17. Sanchez-Lengeling B, Outeiral C, Guimaraes GL, et al. Optimizing Distributions over Molecular Space. An Objective-Reinforced Generative Adversarial Network for Inverse-Design Chemistry (ORGANIC). 2017 August 18;
18. Lim J, Ryu S, Kim JW, et al. Molecular Generative Model Based on Conditional Variational Autoencoder for de Novo Molecular Design. *J. Cheminform.* 2018 December 1; *10* (1):1–9.
19. Kang S, Cho K. Conditional Molecular Design with Deep Generative Models. *J. Chem. Inf. Model.* 2019 January 28; *59* (1):43–52.
20. Bellman R. A Markovian Decision Process. *J. Math. Mech.* 1957 Jan 1; *6*(5):679-84.
21. Munos R. Error Bounds for Approximate Value Iteration. In *Proceedings of the 20th National Conference on Artificial Intelligence - Volume 2*; AAAI'05; AAAI Press, 2005; pp 1006–1011.
22. Szepesvári C, Munos R. Finite Time Bounds for Sampling Based Fitted Value Iteration. *ICML 2005 - Proc. 22nd Int. Conf. Mach. Learn.* 2005;881–888.
23. Gordon GJ. *Approximate Solutions to Markov Decision Processes*; Carnegie Mellon University, 1999.
24. Sutton RS, Barto AG. *Reinforcement Learning: An Introduction*; MIT press, 2018.
25. Ormoneit D, Glynn P. Kernel-Based Reinforcement Learning in Average-Cost Problems. *IEEE Trans. Automat. Contr.* 2002; *47* (10):1624–1636.
26. Ernst D, Geurts P, Wehenkel L. Tree-Based Batch Mode Reinforcement Learning. *J. Mach. Learn. Res.* 2005; *6*:503–556.
27. Ormoneit D, Sen Ś. Kernel-Based Reinforcement Learning. *Mach. Learn.* 2002; *49* (2):161–178.
28. Shen J, Yang LF. Theoretically Principled Deep RL Acceleration via Nearest Neighbor Function Approximation. In *Proceedings of the AAAI Conference on Artificial Intelligence*; 2021; Vol. 35, pp 9558–9566.
29. Mnih V, Kavukcuoglu K, Silver D, et al. Human-Level Control through Deep Reinforcement Learning. *Nat. 2015 5187540* 2015 February 25; *518* (7540):529–533.
30. Watkins CJCH, Dayan P. Q-Learning. *Mach. Learn.* 1992; *8* (3–4):279–292.
31. Hasselt H. Double Q-Learning. *Adv. Neural Inf. Process. Syst.* 2010; *23*:2613–2621.
32. Schaul T, Quan J, Antonoglou I, et al. Prioritized Experience Replay. *arXiv Prepr.*

arXiv1511.05952 2015;

33. Dabney W, Rowland M, Bellemare MG, et al. Distributional Reinforcement Learning with Quantile Regression. In *Thirty-Second AAAI Conference on Artificial Intelligence*; 2018.
34. Wang Z, Schaul T, Hessel M, et al. Dueling Network Architectures for Deep Reinforcement Learning. In *International conference on machine learning*; 2016; pp 1995–2003.
35. Williams RJ. Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning. *Mach. Learn.* 1992; *8* (3):229–256.
36. Ghavamzadeh M, Mahadevan S. Hierarchical Policy Gradient Algorithms. *Comput. Sci. Dep. Fac. Publ. Ser.* 2003;173.
37. Greensmith E, Bartlett PL, Baxter J. Variance Reduction Techniques for Gradient Estimates in Reinforcement Learning. *J. Mach. Learn. Res.* 2004; *5* (9):.
38. Konda VR, Tsitsiklis JN. Actor-Critic Algorithms. In *Advances in neural information processing systems*; 2000; pp 1008–1014.
39. Degris T, Pilarski PM, Sutton RS. Model-Free Reinforcement Learning with Continuous Action in Practice. In *2012 American Control Conference (ACC)*; 2012; pp 2177–2182.
40. Mnih V, Badia AP, Mirza M, et al. Asynchronous Methods for Deep Reinforcement Learning. In *International conference on machine learning*; 2016; pp 1928–1937.
41. Schulman J, Moritz P, Levine S, et al. High-Dimensional Continuous Control Using Generalized Advantage Estimation. *arXiv Prepr. arXiv1506.02438* 2015;
42. Olivecrona M, Blaschke T, Engkvist O, et al. Molecular De-Novo Design through Deep Reinforcement Learning. *J. Cheminform.* 2017 September 4; *9* (1):1–14.
43. Lu F, Li M, Min X, et al. De Novo Generation of Dual-Target Ligands Using Adversarial Training and Reinforcement Learning. *Brief. Bioinform.* 2021 November 5; *22* (6):.
44. You J, Liu B, Ying R, et al. Graph Convolutional Policy Network for Goal-Directed Molecular Graph Generation. *Adv. Neural Inf. Process. Syst.* 2018 June 7; *2018-Decem*:6410–6421.
45. Popova M, Shvets M, Oliva J, et al. MolecularRNN: Generating Realistic Molecular Graphs with Optimized Properties. *arXiv Prepr. arXiv1905.13372* 2019;
46. Simm GNC, Pinsler R, Hernández-Lobato JM. Reinforcement Learning for Molecular Design Guided by Quantum Mechanics. *37th Int. Conf. Mach. Learn. ICML 2020* 2020 February 18; *PartF16814*:8906–8916.
47. Guo J, Fialková V, Arango JD, et al. Improving De Novo Molecular Design with Curriculum Learning. 2021 October 20;
48. Zhou Z, Kearnes S, Li L, et al. Optimization of Molecules via Deep Reinforcement Learning. *Sci. Rep.* 2019 July 24; *9* (1):1–10.
49. Tang B, He F, Liu D, et al. AI-Aided Design of Novel Targeted Covalent Inhibitors against SARS-CoV-2. *bioRxiv Prepr. Serv. Biol.* 2020;
50. Jumper J, Evans R, Pritzel A, et al. Highly Accurate Protein Structure Prediction with AlphaFold. *Nat. 2021 5967873* 2021 July 15; *596* (7873):583–589.
51. Wen J, Kumar S, Gummadi R, et al. Characterizing the Gap Between Actor-Critic and Policy Gradient. *arXiv Prepr. arXiv2106.06932* 2021;
52. Popova M, Isayev O, Tropsha A. Deep Reinforcement Learning for de Novo Drug Design. *Sci. Adv.* 2018 July 25; *4* (7):.

53. Wang J, Hsieh CY, Wang M, et al. Multi-Constraint Molecular Generation Based on Conditional Transformer, Knowledge Distillation and Reinforcement Learning. *Nat. Mach. Intell. 2021 310* 2021 October 18; *3* (10):914–922.

54. Cai T, Xie L, Chen M, et al. Exploration of Dark Chemical Genomics Space via Portal Learning: Applied to Targeting the Undruggable Genome and COVID-19 Anti-Infective Polypharmacology. 2021 November 23;

55. Scholkopf B, Locatello F, Bauer S, et al. Towards Causal Representation Learning. *Proc. IEEE* 2021 February 22; *109* (5):612–634.

56. Ghosh D, Rahme J, Kumar A, et al. Why Generalization in RL Is Difficult: Epistemic POMDPs and Implicit Partial Observability. *Adv. Neural Inf. Process. Syst.* 2021; *34*:.

57. Zhou Z, Kearnes S, Li L, et al. Optimization of Molecules via Deep Reinforcement Learning. *Sci. Rep.* 2019 July 24; *9* (1):1–10.

58. Agarwal R, Liang C, Schuurmans D, et al. Learning to Generalize from Sparse and Underspecified Rewards. *36th Int. Conf. Mach. Learn. ICML 2019* 2019 February 19; *2019-June*:184–194.

59. Yang R, Sun X, Narasimhan K. A Generalized Algorithm for Multi-Objective Reinforcement Learning and Policy Adaptation. *Adv. Neural Inf. Process. Syst.* 2019 August 21; *32*:.

60. Chen D, Wang Y, Gao W. A Two-Stage Multi-Objective Deep Reinforcement Learning Framework. *Front. Artif. Intell. Appl.* 2020 August 24; *325*:1063–1070.

61. Cai T, Lim H, Abbu KA, et al. MSA-Regularized Protein Sequence Transformer toward Predicting Genome-Wide Chemical-Protein Interactions: Application to GPCRome Deorphanization. *J. Chem. Inf. Model.* 2021 April 26; *61* (4):1570–1582.

62. Cai T, Abbu KA, Liu Y, et al. DeepREAL: A Deep Learning Powered Multi-Scale Modeling Framework Towards Predicting Out-of-Distribution Receptor Activity of Ligand Binding. *bioRxiv* 2021 September 15;2021.09.12.460001.

63. Mistry J, Chuguransky S, Williams L, et al. Pfam: The Protein Families Database in 2021. *Nucleic Acids Res.* 2021 January 8; *49* (D1):D412–D419.

64. Liu Y, Wu Y, Shen X, et al. COVID-19 Multi-Targeted Drug Repurposing Using Few-Shot Learning. *Front. Bioinforma.* 2021 June 15; *0*:18.

65. Liu Q, Qiu Y, Xie L. Predictive Modeling of Multiplex Chemical Phenomics for Novel Cells and Patients: Applied to Personalized Alzheimer's Disease Drug Repurposing. *bioRxiv* 2021 January 1;2021.08.09.455708.

66. He D, Liu Q, Xie L. Robust Prediction of Patient-Specific Clinical Response to Unseen Drugs From in Vitro Screens Using Context-Aware Deconfounding Autoencoder. *bioRxiv* 2021;

67. Marx V. Biology Begins to Tangle with Quantum Computing. *Nat. Methods 2021 187* 2021 June 23; *18* (7):715–719.

68. Grinter SZ, Zou X. Challenges, Applications, and Recent Advances of Protein-Ligand Docking in Structure-Based Drug Design. *Molecules* 2014; *19* (7):10150–10176.

69. Jaiteh M, Rodríguez-Espigares I, Selent J, et al. Performance of Virtual Screening against GPCR Homology Models: Impact of Template Selection and Treatment of Binding Site Plasticity. *PLOS Comput. Biol.* 2020; *16* (3):e1007680.