# Batch Reinforcement Learning in the Real World: A Survey

**Yuwei Fu, Wu Di, Benoit Boulet**
Dept. of Electrical and Computer Engineering
McGill University
{yuwei.fu, di.wu5, benoit.boulet}@mail.mcgill.ca

## Abstract

Reinforcement learning (RL) aims to learn an optimal control by interacting with the environments. Reinforcement learning, especially deep reinforcement learning, has developed rapidly in the last few years. However, in the real world, it can be very difficult or even impossible to have a suitable simulator for the RL agent to interact with. Meanwhile, training an RL agent by directly interacting with the real-world environment may lead the system to some dangerous states. Due to these concerns, it is hard to directly apply classical online reinforcement learning algorithms for real-world applications. Batch reinforcement learning, which can learn the control policy from a given dataset, holds the promise to tackle real-world problems. There is a lack of work that introduces batch RL applications in real-world problems. In this paper, we reviewed examples of applying batch RL to various real-world applications. By doing this survey, we hope to provide a series of examples of using batch RL for real-world applications for both the researchers and engineers.

## 1 Introduction

Reinforcement learning (RL) is a powerful paradigm for control problems through trial-and-error learning [84]. Incorporated with latent representations learned from DNNs, recent years have seen a surge of deep RL methods that attain super-human performance in a wide range of domains [61, 79, 2]. However, most of these methods use games [5] or simulated environments [87] as testbeds, which usually require millions of samples to learn an optimal policy. On the other hand, real life is different from games where we can efficiently generate a large amount of data and restart again whenever we fail. For many real-world applications, not only there is no existing simulator, but building one is particularly hard due to the complex unknown system dynamics [18]. Without access to the environment simulator means acting and exploring must be done using data from the real system, which can be time-consuming, expensive or dangerous in many scenarios [26], such as manufacturing, healthcare, and robotics.

Batch RL, also known as offline RL, refers to the problem of learning a near-optimal policy from a given and fixed dataset without additional interaction with the environment [48, 52]. In addition to the *pure batch setting*, early work mostly focused on a variant of the *growing batch setting*, in which the agent is allowed to occasionally interact with the environment to build the batch incrementally [49], as illustrated in Figure 1. Batch RL is viewed as one of the most promising methods for successful real-world RL applications for several reasons: (1) Batch RL allows us to leverage existing datasets for many real-world problems, in which collecting online data is costly, time-consuming or dangerous [98, 13, 101]. (2) Batch RL methods have shown better sample efficiency for many challenging problems [20, 8]. (3) Some advanced supervised learning approaches can be easily transferred to the

Offline Reinforcement Learning Workshop at Neural Information Processing Systems, 2020.

batch RL setting [66]. (4) Many batch RL algorithms are model-free methods that do not need the knowledge of the system dynamics [6].

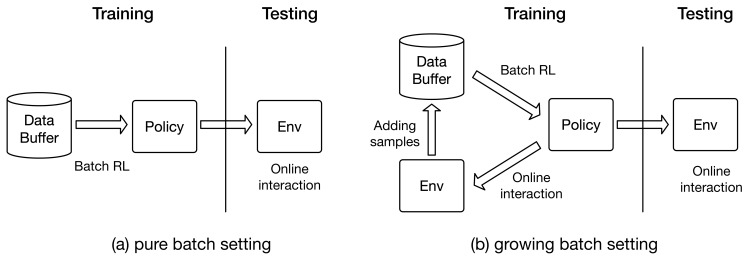


Figure 1: Illustration of different batch RL settings. (a) In the *pure batch setting*, an agent learns a policy purely from the pre-collected dataset without any interaction with the environment during training. (b) In the *growing batch setting*, the agent is allowed to occasionally interact with the environment with latest policy to build the batch incrementally.

In the last two decades [19, 71], various batch RL algorithms have been proposed, and showed promising results in different real-world applications, such as robot control [43], medical treatment [21] or energy system management [27]. However, most of the prior works used simple function approximators with low-dimensional input features, which limited their applications to problems with relatively small state spaces. On the other hand, some recent works [24, 44, 95] have shown that the standard off-policy deep RL methods fail under the batch setting due to the distribution mismatch. Different solutions have been proposed to mitigate this problem, such as adding constraints to the policy [95], using ensembles [1], imitating good actions [92], learning a dynamics model [100], or learning conservative value functions [45].

However, we are facing a gap between current research work and real-world applications. Existing surveys on batch RL such as Lange et al. [48] and Levine et al. [52] focused mainly on introducing different algorithms. A clear summarization of how to apply batch RL in different real-world problems has not been well presented. To connect the theory with real-world problems, this paper reviewed different examples of applying batch RL in various real-world scenarios across the last two decades. By doing this survey, we hope to provide realistic examples to guide researchers developing more practical algorithms. This paper is organized as follows: we start with a general overview with different batch RL methods in Section 2. In Section 3, we review some examples of applying batch RL in different real-world applications. In Section 4, we summarize some challenges and open problems of current batch RL methods, and Section 5 concludes this paper.

## 2 Background

Batch RL considers the problem of learning a policy $\pi(a|s)$ from a fixed dataset $\mathcal{D}$ consisting of single-step transitions $(s, a, r, s')$ [48]. Furthermore, the dataset can be collected by agents with different policies from different control tasks, including non-RL policies, such as human demonstrations [23]. In batch RL, we ignore the exploration problem and focus on exploiting the fixed dataset.

### 2.1 Classic Batch RL Methods

**FQI.** Fitted Q-Iteration (FQI) [19] is one of the most representative batch RL algorithms, which learns an approximation of the optimal action-value function $Q^*$ from a finite set of experience samples $\mathcal{B} = \{(s_i, a_i, r_i, s'_i)\}$. FQI is derived from the *fitted value iteration* approach [28], and reformulates the origin RL problem as a sequence of supervised regression problems (Algorithm 1). At each iteration, FQI computes the TD error to update the Q function with all samples in the dataset. The batch approach allows us to use any regression algorithms, such as tree-based regressor [19], kernel-based regressor [64] or neural networks [71], and prior studies have proved FQI to be sample-efficient to exploit the information contained in the collected samples.

**NFQ & DFQ.** Neural Fitted Q-Iteration (NFQ) [71] is a realization of fitted Q-Iteration where multi-layer perceptrons are used to approximate the Q function. NFQ usually uses a shallow neural network, *e.g.*, 2 layer MLP with less than 10 neurons, as the function approximator, and has been

**Algorithm 1:** Fitted Q-Iteration

**Input:** collected transition samples $\mathcal{B}$, a regression algorithm $f_\theta$.
Initialize Q-value function $\hat{Q}_0(s,a) = 0$.
**for** step $t$ in $\{1, \cdots, N\}$ **do**
    Evaluate $\hat{Q}_{t-1}(s,a)$ for all state-action pairs.
    Compute target value for each sample.
        $y(s,a) = r(s,a) + \gamma \max_{a'} \hat{Q}_{t-1}(s',a')$.
    Run regression algorithm $f_\theta$ on dataset $\{(s,a), y(s,a)\}$ to get $\hat{Q}_t(s,a)$.
**end for**

shown to generalize well on some simple control tasks with only a few training examples. Besides, NFQ proposed a "growing batch" variant of batch RL, that relaxes the "pure batch" setting and allows the agent to interact with the environment to extend the set of sample experience. Most of the early batch RL methods are limited to problems with low dimensionality, Deep Fitted Q-Iteration (DFQ) [49] solves this problem by using a deep auto-encoder model to extract low dimensional embeddings $\phi(s)$ from high dimensional visual inputs $s$, and then apply FQI to the set of experience samples $\{(\phi(s), a, r, \phi(s'))\}$ to learn state embeddings and control policies at the same time. FQI, NFQ, and DFQ laid the ground for many current modern Deep RL methods such as DQN and its variants [61, 54, 35].

### 2.2 DRL-based Batch RL Methods

**Model-free approaches.** Recent work [24, 52] reports that directly applying existing value-based off-policy DRL algorithms in an offline setting usually results in poor performance. [24, 44, 50] attribute this issue to out-of-distribution actions and overfitting caused by the distribution mismatch between the offline training data and the learned policy. To remedy this problem, many heuristic approaches are proposed to regularize the learned policy to keep close to the behavioral policy. For example, BCQ [24] uses a variational auto-encoder (VAE) to generate actions that are similar to samples in the buffer. SPIBB [50] restricts the support of the learned policy to the support of the behavior policy for safe policy improvement. KL-Control [37] uses KL divergence, and BEAR [44] uses maximum mean discrepancy (MMD) divergence [29] as a regularizer to the loss function. [95] proposed a general framework, named BRAC, and evaluate the effectiveness of different components in the existing methods.

Besides the behavior regularized methods, some recent work solves batch RL problems with weighted behavior cloning, which uses an estimate of the advantage function [63, 92] to select samples. AWR [66] uses advantage-weighted regression to convert batch RL to a standard supervised learning problem. ABM [78] biases the RL policies to promising executed actions. [93] proposed *critic-regularized regression* (CRR) that reduce batch RL to a form of value-filtered regression. In addition, CQL [45] learns a conservative Q-function to directly tackle the overestimation problem. [55] provided a stronger theoretical performance guarantee by modifying the Bellman optimality back-up to take a more conservative update. BRPO [83] learns residual policies and maximizes a conservative lower bound on the policy performance. [7] further proposed a general framework to unify existing approaches and introduce two families of pessimistic algorithms. Another recent work [1] demonstrates that simple ensemble methods with large and diverse datasets allow us to learn high quality polices.

**Model-based approaches.** Another line of research focuses on incorporating model-based RL methods into the batch RL setting [3, 41, 100, 58]. Model-based RL algorithms can learn policies with out-of-distribution states, which are absent in model-free batch RL methods. MoREL [41] first learns a pessimistic MDP with the fixed dataset to avoid model exploitation problem, and then learns a near-optimal policy in this pessimistic MDP. MOPO [100] introduces a penalized MDP framework that adds a penalty to the reward function, and MBOP [3] leverages MPC to learn a goal or reward-conditioned policy.

## 3 Real-world Applications

This section introduces some examples of applying batch RL in various real-world scenarios.

### 3.1 Robotics

Robots have been deployed extensively in production to automate manual tasks to improve efficiency and reduce costs [43]. RL provides a powerful framework for learning autonomous robots that improve continuously through experiences. However, collecting a large amount of data with real robots may be very time consuming or dangerous [57]. In addition, robots suffer from the problem of wear and tear and are usually costly to repair [2], the ability to learn from fixed dataset makes batch RL an appealing method for robotics applications. Moreover, unlike previous control-based solutions that require the knowledge of the system dynamics, model-free batch RL algorithms allow us to learn control policies purely from collected samples [6].

A notable example is the RoboCup soccer leagues [42], where two teams of mobile robots compete in a soccer game. [72] developed NFQ-based soccer robots to learn various mobile control skills and won several world championships. Experiments showed that a two-layer MLP with 10 neurons for each hidden layer was good enough to learn effective policies in both simulation and real soccer robots. Later, [11] proposed Q-Batch which uses an n-step return to exploit the episodic nature of the interactions. Other examples include [6], which used NFQ to solve a robot control problem in the classical control theory of following a reference state (*e.g.*, position set point), and the negative reward is set to be the velocity difference between the state and the reference. [8] introduced a *data-driven* approach to applying DRL to learn to perform manipulation tasks on real robots from vision. The key component is an interactive approach, called *reward sketching*, to elicit human preferences to learn the reward function for a new task. The learned reward function is later used to annotate all historical data. which is used to learn the control policy with batch RL. After the retrospective reward assignment, a variant of D4PG [4] is used as the batch RL algorithm to learn control policy purely from the annotated dataset without online interaction.

### 3.2 Dialogue Systems

A dialogue system aims to build chatbots that can communicate with humans through natural language [81, 53, 94] to solve different tasks, *i.e.*, restaurant booking, and GPS navigation. Dialogue systems have been an active research area in Human-Computer Interaction (HCI) for decades, and it is now prevalent in different commercial applications on various digital devices, such as Apple's Siri, Microsoft's Cortana, and Amazon's Alexa [40, 56]. Most early work resorts to hand-crafted rules and templates for simple Q&A applications. Later, various approaches try to solve this problem by learning a statistic language model from data [39]. In recent years, many neural dialogue generation models following the seq-to-seq framework with attention mechanisms have achieved huge progress [90, 60]. However, a seq-to-seq based model trained with the MLE objective function usually tends to generate highly generic responses and fails to account for long-term influence [53].

To overcome these issues, there is a growing number of methods that use RL to learn a stochastic dialogue generation policy [53, 14]. Here, we introduce some of these methods that use batch RL to learn from fixed corpora. [67] proved batch RL methods, such as FQI and LSPI [47], are sample efficient to learn near-optimal policies from a few hundreds of simulated dialogue examples. [34] proposed a hybrid method that combines batch RL and supervised learning to learn dialogue policies from a fixed dataset. [37] proposed KL-Control, which leverages a pre-trained language model as prior $p(a|s)$ to generate batch data via human interaction and then uses KL divergence to constrain the learned policy to stay close to the prior. Moreover, KL-Control designs different intrinsic reward functions to capture implicit human preferences in dialog, such as sentiment, questions, and laughters.

### 3.3 Energy Management System

To prevent the next energy crisis and protect our environment, many countries continue to increase their budgets for renewable energy investment, such as solar, geothermal, and wind power, in the last few years [10]. However, harnessing energy from renewable sources is a challenging task, because the power generation process is usually decentralized, intermittent, and uncontrollable while the consumers can use electricity in arbitrary quantities at any time [76]. Thus, modern power systems require more advanced control techniques for power generation, distribution and storage with higher reliability and better security [27]. Meanwhile, with the popularization of digital metering and communication infrastructures, the power system generates unprecedented high volumes of data to make it possible for us to use data-driven machine learning algorithms to solve these problems.

Without the knowledge of the system dynamics, model-free batch RL methods have been widely used in this domain to learn complex control policies.

We start with the example of smart grids. The smart grid is an advanced digital two-way power flow system, which provides communication between consumers and suppliers [46]. It aims to distribute power to consumers more efficiently, and economically by adaptive adjustment according to supply-demand mismatches [15]. [20] compared FQI and MPC methods on an electric power system problem and suggest a combination of FQI and MPC when a good model is available. [75] applied FQI to learn to schedule a cluster of domestic electric water heaters under a time-varying electricity price to minimize electricity cost in a smart grid environment. [73] later extended FQI to the residential demand response scenario, such as heat-pump thermostat, which incorporates a forecast of the exogenous data. A known price profile is used to compute reward and the objective is to minimize the gap between the day-ahead consumption plan and the actual consumption. Later [74] further extended this method by incorporating an auto-encoder network to learn compact state representation for FQI. [59] used FQI to learn battery energy management policies, which are tested by simulation in a residential setting using data from Belgian residential consumers.

In addition to the applications in smart grids, [62] applied DFQ to control space heating in buildings. [97] used LSPI-based batch RL method and a virtual transition generator to solve the problem of setting the tap position of load tap changers (LTCs) for voltage regulation in radial power distribution systems. [9] used FQI to learn an optimal cost-reducing charging policy for individual plug-in electric vehicle (PEV) from a collection of transition samples. A Bayesian neural network is employed to predict the second-day electricity price based on historical data.

### 3.4 Healthcare

AI for healthcare is regarded as one of the most promising areas in which AI can improve our lives [38]. The increasing availability of electronic health data paves the way for applying data-driven machine learning approaches to solve various challenging tasks, and recent years have seen a number of successful applications of AI in healthcare, such as disease diagnosis, medical imaging, drug development, critical care, treatment recommendation, and epidemic prediction [99]. Batch RL is particularly attractive for some healthcare tasks for two reasons. First, many healthcare problems involve making a sequence of decisions with a long-term consequence that can be formulated in an RL setting, for example, some drugs appear to work after a while. Second, patient safety is always the first priority, and thus we can hardly collect enough online samples and need to learn from existing offline samples.

[21] applied FQI to learn optimal structured treatment interruption strategies for HIV infected patients directly from simulated clinical data, where the reward is defined as the equilibrium point between healthy and unhealthy blood cells in the patient. [77] used FQI to optimize the treatment choices for schizophrenia with a bootstrap confidence interval. [82] applied FQI to learn an intervention strategy directly from gene expression data. [22] applied FQI to the optimization of anemia treatment. [70] used Dueling Double-DQN to deduce optimal treatment policies for patients with sepsis, in which a sparse auto-encoder is used to learn continuous features. SOFA score and patient's lactate levels are used as reward functions to indicate the goodness of patient's health. [69] explored the use of FQI to learn ventilator weaning policy from examples in historical ICU data with a complicated hand-designed reward.

### 3.5 Manufacturing

Batch RL is also widely used in real manufacturing problems [96]. Ramp-up is an important concept for efficient manufacturing systems, which refers to the period between completion of development and full capacity utilization [86]. Traditionally, ramp-up process largely depends on expert knowledge and is difficult to solve with a machine learning algorithm due to the limited similarity between different systems. [16] formulated the ramp-up process as an MDP, and applied a Q-learning based batch RL method to reduce the ramp-up time. [25] formalized job-shop scheduling problems as multi-agent MDP, and learned an optimal scheduling policy with NFQ. Besides, [32] provided an open benchmark, named Industrial Benchmark, for comparing different batch RL algorithms in different industrial scenarios [33].

### 3.6 Others

In addition to the applications outlined above, some prior work also applies batch RL in all kinds of other real-world problems. [36] proposed a batch RL method, called *parameterized batch actor-critic* (PBAC), for longitudinal control of autonomous land vehicles (ALVs). PBAC learns parameterized kernel features from collected samples to approximate the value functions and policies. The actor and critic are updated using least-squares-based batch updating rules and the reward is a combination of speed tracking error and energy cost. [31] introduced DRIFT, an efficient software testing framework, which formalizes the software testing task as an MDP. In DRIFT, the state is a tree that corresponds to a hierarchy of GUI elements, action is the possible user interaction, and the reward is generated by counting the number of times that a key function is triggered. To avoid the cost of data collection, DRIFT is trained from existing data with DQN in a batch RL paradigm.

## 4 Challenges and Open Problems

In the previous section, we have seen many successful examples of applying batch RL in real-world problems. However, several practical challenges remain to be solved for a wide array of large-scale real-world RL applications. For example, high-dimensional continuous action or state spaces, strict safety constraints and non-stationary dynamics [18]. Here, we discuss three particular challenges.

### 4.1 Challenges to Future Progress

**Effective offline evaluation protocols.** A successful deployment of RL to real production systems relies on robust evaluation, and most of the existing batch RL algorithms are evaluated online by interacting with an environment simulator. However, in many real-world applications, such as the recommender system [85], it is impractical to tune hyperparameters on the real system. Therefore, we need to find a more effective offline evaluation protocol, *i.e.*, using *off-policy policy evaluation* (OPE) methods, to estimate the policy's performance purely from logged data [91]. [30, 23] apply a naïve approach that tunes hyperparameters on some selected tasks, but this simple method does not work in real-world scenarios where we only have a single task. Recently, [65] proposed an offline hyperparameter selection method which uses *fitted Q evaluation* (FQE) [51] to compute scalar statistics without interacting with the environment to rank policies. Furthermore, which OPE methods to use and hyperparameter selection for the OPE methods are also challenging open problems [30].

**Unspecified reward functions.** Another challenge for batch RL in the real-world is the lack of specified reward functions. RL problems are formulated as learning an optimal policy that maximizes the expected cumulative discounted rewards [84]. Unfortunately, there are no existing reward functions in the real-world, and most of the previous methods rely on hand-crafted reward signals, which require expert domain knowledge and careful parameter tuning [6, 89, 37]. In addition, many real-world applications have multi-dimensional goals. For example, we usually want a robot to solve a task without breaking any safety constraints [12] or we want a power grid to reduce costs and improve efficiency [15]. Thus, we need to design more complex multi-objective reward functions or better optimization objectives.

**Off-policy training & deadly triad.** Moreover, most current batch RL methods suffer from the deadly triad [84, 88] of function approximation, off-policy learning, and bootstrapping. Some work provides theoretical analysis in tabular problem settings [24, 44], but real-world applications differ substantially from these assumptions. We can observe that these methods sometimes fail due to divergence [24] which is unacceptable for many real-world problems with critical safety concerns. Given the potential instability, more efforts are expected to achieve safer batch RL.

### 4.2 Open Problems

The evaluation challenge mentioned above highlights the importance for researchers to design batch RL algorithms that are less sensitive to the hyperparameters [95]. Another open problem is how to develop efficient reward specification methods to mitigate the issue of laborious reward engineering [80]. A potential solution to overcome this problem is to incorporate some general intrinsic rewards into the learning process [68]. Batch RL is promising for sample-efficient practical applications with safety concerns, however, we need more theoretical work to lay the ground for more reliable real-world deployments [55]. In addition, having large datasets and open benchmarks has proved

to be beneficial for the development of different machine learning domains. Hence, we are looking forward to more open benchmarks, such as *realworldrl-suite* [17], *rl-unplugged* [30] and *d4rl* [23], to accelerate progress in this emerging field.

## 5 Conclusion

To connect current research work on batch RL with real-world applications, in this paper, we reviewed some examples of applying batch RL in various real-world problems across the last two decades. We highlighted some challenges for real-world batch RL, such as offline evaluation, unspecified reward functions, and the deadly triad of off-policy learning. With the development of more advanced batch RL algorithms and the increasing amount of real-world data, batch RL is promising to solve more challenging real-world tasks. A potential future direction is to transfer samples or trajectories from different tasks to further improve the sample efficiency. Another promising future direction is to active select samples form the large fixed dataset to accelerate learning.

## References


[1] Rishabh Agarwal, Dale Schuurmans, and Mohammad Norouzi. An optimistic perspective on offline reinforcement learning. In *International Conference on Machine Learning*, 2020.

[2] OpenAI: Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Jozefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, et al. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1):3–20, 2020.

[3] Arthur Argenson and Gabriel Dulac-Arnold. Model-based offline planning. *arXiv preprint arXiv:2008.05556*, 2020.

[4] Gabriel Barth-Maron, Matthew W Hoffman, David Budden, Will Dabney, Dan Horgan, Dhruva Tb, Alistair Muldal, Nicolas Heess, and Timothy Lillicrap. Distributed distributional deterministic policy gradients. *arXiv preprint arXiv:1804.08617*, 2018.

[5] Marc G Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013.

[6] Andrea Bonarini, Claudio Caccia, Alessandro Lazaric, and Marcello Restelli. Batch reinforcement learning for controlling a mobile wheeled pendulum robot. In *IFIP International Conference on Artificial Intelligence in Theory and Practice*, pages 151–160. Springer, 2008.

[7] Jacob Buckman, Carles Gelada, and Marc G Bellemare. The importance of pessimism in fixed-dataset policy optimization. *arXiv preprint arXiv:2009.06799*, 2020.

[8] Serkan Cabi, Sergio Gómez Colmenarejo, Alexander Novikov, Ksenia Konyushkova, Scott Reed, Rae Jeong, Konrad Zolna, Yusuf Aytar, David Budden, Mel Vecerik, et al. Scaling data-driven robotics with reward sketching and batch reinforcement learning. *arXiv*, pages arXiv–1909, 2019.

[9] Adriana Chiş, Jarmo Lundén, and Visa Koivunen. Reinforcement learning-based plug-in electric vehicle charging with forecasted price. *IEEE Transactions on Vehicular Technology*, 66(5):3674–3684, 2016.

[10] Steven Chu, Yi Cui, and Nian Liu. The path towards sustainable energy. *Nature materials*, 16(1):16–22, 2017.

[11] Joao Cunha, Rui Serra, Nuno Lau, Luís Seabra Lopes, and Antóio JR Neves. Batch reinforcement learning for robotic soccer using the q-batch update-rule. *Journal of Intelligent & Robotic Systems*, 80(3-4):385–399, 2015.

[12] Gal Dalal, Krishnamurthy Dvijotham, Matej Vecerik, Todd Hester, Cosmin Paduraru, and Yuval Tassa. Safe exploration in continuous action spaces. *arXiv preprint arXiv:1801.08757*, 2018.

[13] Sudeep Dasari, Frederik Ebert, Stephen Tian, Suraj Nair, Bernadette Bucher, Karl Schmeckpeper, Siddharth Singh, Sergey Levine, and Chelsea Finn. Robonet: Large-scale multi-robot learning. *arXiv preprint arXiv:1910.11215*, 2019.

[14] Bhuwan Dhingra, Lihong Li, Xiujun Li, Jianfeng Gao, Yun-Nung Chen, Faisal Ahmed, and Li Deng. Towards end-to-end reinforcement learning of dialogue agents for information access. *arXiv preprint arXiv:1609.00777*, 2016.

[15] G Dileep. A survey on smart grid technologies and applications. *Renewable Energy*, 146:2589–2625, 2020.

[16] Stefanos Doltsinis, Pedro Ferreira, and Niels Lohse. Reinforcement learning for production ramp-up: A q-batch learning approach. In *2012 11th International Conference on Machine Learning and Applications*, volume 1, pages 610–615. IEEE, 2012.

[17] Gabriel Dulac-Arnold, Nir Levine, Daniel J Mankowitz, Jerry Li, Cosmin Paduraru, Sven Gowal, and Todd Hester. An empirical investigation of the challenges of real-world reinforcement learning. *arXiv preprint arXiv:2003.11881*, 2020.

[18] Gabriel Dulac-Arnold, Daniel Mankowitz, and Todd Hester. Challenges of real-world reinforcement learning. *arXiv preprint arXiv:1904.12901*, 2019.

[19] Damien Ernst, Pierre Geurts, and Louis Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6(Apr):503–556, 2005.

[20] Damien Ernst, Mevludin Glavic, Florin Capitanescu, and Louis Wehenkel. Reinforcement learning versus model predictive control: a comparison on a power system problem. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(2):517–529, 2009.

[21] Damien Ernst, Guy-Bart Stan, Jorge Goncalves, and Louis Wehenkel. Clinical data based optimal sti strategies for hiv: a reinforcement learning approach. In *Proceedings of the 45th IEEE Conference on Decision and Control*, pages 667–672. IEEE, 2006.

[22] Pablo Escandell-Montero, Milena Chermisi, Jose M Martinez-Martinez, Juan Gomez-Sanchis, Carlo Barbieri, Emilio Soria-Olivas, Flavio Mari, Joan Vila-Francés, Andrea Stopper, Emanuele Gatti, et al. Optimization of anemia treatment in hemodialysis patients via reinforcement learning. *Artificial intelligence in medicine*, 62(1):47–60, 2014.

[23] Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

[24] Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, pages 2052–2062, 2019.

[25] Thomas Gabel and Martin Riedmiller. Adaptive reactive job-shop scheduling with reinforcement learning agents. *International Journal of Information Technology and Intelligent Computing*, 24(4):14–18, 2008.

[26] Javier Garcıa and Fernando Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(1):1437–1480, 2015.

[27] Mevludin Glavic, Raphaël Fonteneau, and Damien Ernst. Reinforcement learning for electric power system decision and control: Past considerations and perspectives. *IFAC-PapersOnLine*, 50(1):6918–6927, 2017.

[28] Geoffrey J Gordon. Approximate solutions to markov decision processes. Technical report, CARNEGIE-MELLON UNIV PITTSBURGH PA SCHOOL OF COMPUTER SCIENCE, 1999.

[29] Arthur Gretton, Karsten M Borgwardt, Malte Rasch, Bernhard Schölkopf, and Alexander J Smola. A kernel approach to comparing distributions. In *Proceedings of the National Conference on Artificial Intelligence*, volume 22, page 1637. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2007.

[30] Caglar Gulcehre, Ziyu Wang, Alexander Novikov, Tom Le Paine, Sergio Gómez Colmenarejo, Konrad Zolna, Rishabh Agarwal, Josh Merel, Daniel Mankowitz, Cosmin Paduraru, et al. Rl unplugged: Benchmarks for offline reinforcement learning. *arXiv preprint arXiv:2006.13888*, 2020.

[31] Luke Harries, Rebekah Storan Clarke, Timothy Chapman, Swamy VPLN Nallamalli, Levent Ozgur, Shuktika Jain, Alex Leung, Steve Lim, Aaron Dietrich, José Miguel Hernández-Lobato, et al. Drift: Deep reinforcement learning for functional software testing. *arXiv preprint arXiv:2007.08220*, 2020.

[32] Daniel Hein, Alexander Hentschel, Volkmar Sterzing, Michel Tokic, and Steffen Udluft. Introduction to the" industrial benchmark". *arXiv preprint arXiv:1610.03793*, 2016.

[33] Daniel Hein, Steffen Udluft, Michel Tokic, Alexander Hentschel, Thomas A Runkler, and Volkmar Sterzing. Batch reinforcement learning on the industrial benchmark: First experiences. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 4214–4221. IEEE, 2017.

[34] James Henderson, Oliver Lemon, and Kallirroi Georgila. Hybrid reinforcement/supervised learning of dialogue policies from fixed data sets. *Computational Linguistics*, 34(4):487–511, 2008.

[35] Matteo Hessel, Joseph Modayil, Hado Van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. Rainbow: Combining improvements in deep reinforcement learning. *arXiv preprint arXiv:1710.02298*, 2017.

[36] Zhenhua Huang, Xin Xu, Haibo He, Jun Tan, and Zhenping Sun. Parameterized batch reinforcement learning for longitudinal control of autonomous land vehicles. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 49(4):730–741, 2017.

[37] Natasha Jaques, Asma Ghandeharioun, Judy Hanwen Shen, Craig Ferguson, Agata Lapedriza, Noah Jones, Shixiang Gu, and Rosalind Picard. Way off-policy batch deep reinforcement learning of implicit human preferences in dialog. *arXiv preprint arXiv:1907.00456*, 2019.

[38] Fei Jiang, Yong Jiang, Hui Zhi, Yi Dong, Hao Li, Sufeng Ma, Yilong Wang, Qiang Dong, Haipeng Shen, and Yongjun Wang. Artificial intelligence in healthcare: past, present and future. *Stroke and vascular neurology*, 2(4):230–243, 2017.

[39] Daniel Jurafsky and James H Martin. Speech and language processing: An introduction to speech recognition, computational linguistics and natural language processing. *Upper Saddle River, NJ: Prentice Hall*, 2008.

[40] Veton Kepuska and Gamal Bohouta. Next-generation of virtual personal assistants (microsoft cortana, apple siri, amazon alexa and google home). In *2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*, pages 99–103. IEEE, 2018.

[41] Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. Morel: Model-based offline reinforcement learning. *arXiv preprint arXiv:2005.05951*, 2020.

[42] Hiroaki Kitano, Minoru Asada, Yasuo Kuniyoshi, Itsuki Noda, and Eiichi Osawa. Robocup: The robot world cup initiative. In *Proceedings of the first international conference on Autonomous agents*, pages 340–347, 1997.

[43] Jens Kober, J Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

[44] Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. In *Advances in Neural Information Processing Systems*, pages 11784–11794, 2019.

[45] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *arXiv preprint arXiv:2006.04779*, 2020.

[46] Angeliki Kylili and Paris A. Fokaides. European smart cities: The role of zero energy buildings. *Sustainable Cities and Society*, 15:86 – 95, 2015.

[47] Michail G Lagoudakis and Ronald Parr. Least-squares policy iteration. *Journal of machine learning research*, 4(Dec):1107–1149, 2003.

[48] Sascha Lange, Thomas Gabel, and Martin Riedmiller. Batch reinforcement learning. In *Reinforcement learning*, pages 45–73. Springer, 2012.

[49] Sascha Lange and Martin Riedmiller. Deep auto-encoder neural networks in reinforcement learning. In *The 2010 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2010.

[50] Romain Laroche, Paul Trichelair, and Remi Tachet Des Combes. Safe policy improvement with baseline bootstrapping. In *International Conference on Machine Learning*, pages 3652–3661. PMLR, 2019.

[51] Hoang M Le, Cameron Voloshin, and Yisong Yue. Batch policy learning under constraints. *arXiv preprint arXiv:1903.08738*, 2019.

[52] Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

[53] Jiwei Li, Will Monroe, Alan Ritter, Michel Galley, Jianfeng Gao, and Dan Jurafsky. Deep reinforcement learning for dialogue generation. *arXiv preprint arXiv:1606.01541*, 2016.

[54] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[55] Yao Liu, Adith Swaminathan, Alekh Agarwal, and Emma Brunskill. Provably good batch reinforcement learning without great exploration. *arXiv preprint arXiv:2007.08202*, 2020.

[56] Gustavo López, Luis Quesada, and Luis A Guerrero. Alexa vs. siri vs. cortana vs. google assistant: a comparison of speech-based natural user interfaces. In *International Conference on Applied Human Factors and Ergonomics*, pages 241–250. Springer, 2017.

[57] A Rupam Mahmood, Dmytro Korenkevych, Gautham Vasan, William Ma, and James Bergstra. Benchmarking reinforcement learning algorithms on real-world robots. *arXiv preprint arXiv:1809.07731*, 2018.

[58] Tatsuya Matsushima, Hiroki Furuta, Yutaka Matsuo, Ofir Nachum, and Shixiang Gu. Deployment-efficient reinforcement learning via model-based offline optimization. *arXiv preprint arXiv:2006.03647*, 2020.

[59] Brida V Mbuwir, Frederik Ruelens, Fred Spiessens, and Geert Deconinck. Battery energy management in a microgrid using batch reinforcement learning. *Energies*, 10(11):1846, 2017.

[60] Hongyuan Mei, Mohit Bansal, and Matthew R Walter. Coherent dialogue with attention-based language models. *arXiv preprint arXiv:1611.06997*, 2016.

[61] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[62] Adam Nagy, Hussain Kazmi, Farah Cheaib, and Johan Driesen. Deep reinforcement learning for optimal control of space heating. *arXiv preprint arXiv:1805.03777*, 2018.

[63] Gerhard Neumann and Jan R Peters. Fitted q-iteration by advantage weighted regression. In *Advances in neural information processing systems*, pages 1177–1184, 2009.

[64] Dirk Ormoneit and Śaunak Sen. Kernel-based reinforcement learning. *Machine learning*, 49(2-3):161–178, 2002.

[65] Tom Le Paine, Cosmin Paduraru, Andrea Michi, Caglar Gulcehre, Konrad Zolna, Alexander Novikov, Ziyu Wang, and Nando de Freitas. Hyperparameter selection for offline reinforcement learning. *arXiv preprint arXiv:2007.09055*, 2020.

[66] Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019.

[67] Olivier Pietquin, Matthieu Geist, Senthilkumar Chandramohan, and Hervé Frezza-Buet. Sample-efficient batch reinforcement learning for dialogue management optimization. *ACM Transactions on Speech and Language Processing (TSLP)*, 7(3):1–21, 2011.

[68] Rémy Portelas, Cédric Colas, Lilian Weng, Katja Hofmann, and Pierre-Yves Oudeyer. Automatic curriculum learning for deep rl: A short survey. *arXiv preprint arXiv:2003.04664*, 2020.

[69] Niranjani Prasad, Li-Fang Cheng, Corey Chivers, Michael Draugelis, and Barbara E Engelhardt. A reinforcement learning approach to weaning of mechanical ventilation in intensive care units. *arXiv preprint arXiv:1704.06300*, 2017.

[70] Aniruddh Raghu, Matthieu Komorowski, Imran Ahmed, Leo Celi, Peter Szolovits, and Marzyeh Ghassemi. Deep reinforcement learning for sepsis treatment. *arXiv preprint arXiv:1711.09602*, 2017.

[71] Martin Riedmiller. Neural fitted q iteration–first experiences with a data efficient neural reinforcement learning method. In *European Conference on Machine Learning*, pages 317–328. Springer, 2005.

[72] Martin Riedmiller, Thomas Gabel, Roland Hafner, and Sascha Lange. Reinforcement learning for robot soccer. *Autonomous Robots*, 27(1):55–73, 2009.

[73] Frederik Ruelens, Bert Claessens, Stijn Vandael, Bart De Schutter, Robert Babuska, and Ronnie Belmans. Residential demand response applications using batch reinforcement learning. *arXiv preprint arXiv:1504.02125*, 2015.

[74] Frederik Ruelens, Bert J Claessens, Salman Quaiyum, Bart De Schutter, R Babuška, and Ronnie Belmans. Reinforcement learning applied to an electric water heater: from theory to practice. *IEEE Transactions on Smart Grid*, 9(4):3792–3800, 2016.

[75] Frederik Ruelens, Bert J Claessens, Stijn Vandael, Sandro Iacovella, Pieter Vingerhoets, and Ronnie Belmans. Demand response of a heterogeneous cluster of electric water heaters using batch reinforcement learning. In *2014 Power Systems Computation Conference*, pages 1–7. IEEE, 2014.

[76] Navin Sharma, Pranshu Sharma, David Irwin, and Prashant Shenoy. Predicting solar generation from weather forecasts using machine learning. In *2011 IEEE international conference on smart grid communications (SmartGridComm)*, pages 528–533. IEEE, 2011.

[77] Susan M Shortreed, Eric Laber, Daniel J Lizotte, T Scott Stroup, Joelle Pineau, and Susan A Murphy. Informing sequential clinical decision-making through reinforcement learning: an empirical study. *Machine learning*, 84(1-2):109–136, 2011.

[78] Noah Y Siegel, Jost Tobias Springenberg, Felix Berkenkamp, Abbas Abdolmaleki, Michael Neunert, Thomas Lampe, Roland Hafner, and Martin Riedmiller. Keep doing what worked: Behavioral modelling priors for offline reinforcement learning. *arXiv preprint arXiv:2002.08396*, 2020.

[79] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

[80] Avi Singh, Larry Yang, Kristian Hartikainen, Chelsea Finn, and Sergey Levine. End-to-end robotic reinforcement learning without reward engineering. *arXiv preprint arXiv:1904.07854*, 2019.

[81] Satinder P Singh, Michael J Kearns, Diane J Litman, and Marilyn A Walker. Reinforcement learning for spoken dialogue systems. In *Advances in Neural Information Processing Systems*, pages 956–962, 2000.

[82] Utku Sirin, Faruk Polat, and Reda Alhajj. Employing batch reinforcement learning to control gene regulation without explicitly constructing gene regulatory networks. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence*, number CONF, pages 2042–2048. AAAI Press, 2013.

[83] Sungryull Sohn, Yinlam Chow, Jayden Ooi, Ofir Nachum, Honglak Lee, Ed Chi, and Craig Boutilier. Brpo: Batch residual policy optimization. *arXiv preprint arXiv:2002.05522*, 2020.

[84] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[85] Adith Swaminathan, Akshay Krishnamurthy, Alekh Agarwal, Miro Dudik, John Langford, Damien Jose, and Imed Zitouni. Off-policy evaluation for slate recommendation. In *Advances in Neural Information Processing Systems*, pages 3632–3642, 2017.

[86] Christian Terwiesch and Roger E Bohn. Learning and process improvement during production ramp-up. *International journal of production economics*, 70(1):1–19, 2001.

[87] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012.

[88] Hado Van Hasselt, Yotam Doron, Florian Strub, Matteo Hessel, Nicolas Sonnerat, and Joseph Modayil. Deep reinforcement learning and the deadly triad. *arXiv preprint arXiv:1812.02648*, 2018.

[89] Mel Vecerik, Todd Hester, Jonathan Scholz, Fumin Wang, Olivier Pietquin, Bilal Piot, Nicolas Heess, Thomas Rothörl, Thomas Lampe, and Martin Riedmiller. Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards. *arXiv preprint arXiv:1707.08817*, 2017.

[90] Oriol Vinyals and Quoc Le. A neural conversational model. *arXiv preprint arXiv:1506.05869*, 2015.

[91] Cameron Voloshin, Hoang M Le, Nan Jiang, and Yisong Yue. Empirical study of off-policy policy evaluation for reinforcement learning. *arXiv preprint arXiv:1911.06854*, 2019.

[92] Qing Wang, Jiechao Xiong, Lei Han, Han Liu, Tong Zhang, et al. Exponentially weighted imitation learning for batched historical data. In *Advances in Neural Information Processing Systems*, pages 6288–6297, 2018.

[93] Ziyu Wang, Alexander Novikov, Konrad Żołna, Jost Tobias Springenberg, Scott Reed, Bobak Shahriari, Noah Siegel, Josh Merel, Caglar Gulcehre, Nicolas Heess, et al. Critic regularized regression. *arXiv preprint arXiv:2006.15134*, 2020.

[94] Gellért Weisz, Paweł Budzianowski, Pei-Hao Su, and Milica Gašić. Sample efficient deep reinforcement learning for dialogue systems with large action spaces. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 26(11):2083–2097, 2018.

[95] Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

[96] AS Xanthopoulos, Athanasios Kiatipis, Dimitris E Koulouriotis, and Sepp Stieger. Reinforcement learning-based and parametric production-maintenance control policies for a deteriorating manufacturing system. *IEEE Access*, 6:576–588, 2017.

[97] Hanchen Xu, Alejandro D Domínguez-García, and Peter W Sauer. Optimal tap setting of voltage regulation transformers using batch reinforcement learning. *IEEE Transactions on Power Systems*, 35(3):1990–2001, 2019.

[98] Fisher Yu, Wenqi Xian, Yingying Chen, Fangchen Liu, Mike Liao, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving video database with scalable annotation tooling. *arXiv preprint arXiv:1805.04687*, 2(5):6, 2018.

[99] Kun-Hsing Yu, Andrew L Beam, and Isaac S Kohane. Artificial intelligence in healthcare. *Nature biomedical engineering*, 2(10):719–731, 2018.

[100] Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. Mopo: Model-based offline policy optimization. *arXiv preprint arXiv:2005.13239*, 2020.

[101] Xiao Yu, Hao Ma, Bo-June Hsu, and Jiawei Han. On building entity recommender systems using user click log and freebase knowledge. In *Proceedings of the 7th ACM international conference on Web search and data mining*, pages 263–272, 2014.