
DEEP REINFORCEMENT LEARNING: AN OVERVIEW

Yuxi Li (yuxili@gmail.com)

ABSTRACT

We give an overview of recent exciting achievements of deep reinforcement learning (RL). We discuss six core elements, six important mechanisms, and twelve applications. We start with background of machine learning, deep learning and reinforcement learning. Next we discuss core RL elements, including value function, in particular, Deep Q-Network (DQN), policy, reward, model, planning, and exploration. After that, we discuss important mechanisms for RL, including attention and memory, in particular, differentiable neural computer (DNC), unsupervised learning, transfer learning, semi-supervised learning, hierarchical RL, and learning to learn. Then we discuss various applications of RL, including games, in particular, AlphaGo, robotics, natural language processing, including dialogue systems (a.k.a. chatbots), machine translation, and text generation, computer vision, neural architecture design, business management, finance, healthcare, Industry 4.0, smart grid, intelligent transportation systems, and computer systems. We mention topics not reviewed yet. After listing a collection of RL resources, we present a brief summary, and close with discussions.

CONTENTS

1	Introduction	5
2	Background	6
2.1	Machine Learning	7
2.2	Deep Learning	8
2.3	Reinforcement Learning	9
2.3.1	Problem Setup	9
2.3.2	Value Function	9
2.3.3	Temporal Difference Learning	9
2.3.4	Multi-step Bootstrapping	10
2.3.5	Function Approximation	11
2.3.6	Policy Optimization	12
2.3.7	Deep Reinforcement Learning	12
2.3.8	RL Parlance	13
2.3.9	Brief Summary	14
3	Core Elements	14
3.1	Value Function	14
3.1.1	Deep Q-Network (DQN)	14
3.1.2	Double DQN	15
3.1.3	Prioritized Experience Replay	15
3.1.4	Dueling Architecture	16
3.1.5	More DQN Extensions	16
3.2	Policy	16
3.2.1	Actor-Critic	16
3.2.2	Policy Gradient	17
3.2.3	Combining Policy Gradient and Off-Policy RL	19
3.3	Reward	19
3.4	Model	20
3.5	Planning	21
3.6	Exploration	21
4	Important Mechanisms	21
4.1	Attention and Memory	21
4.1.1	Differentiable Neural Computer	22
4.2	Unsupervised Learning	22
4.2.1	Horde	22
4.2.2	Unsupervised Auxiliary Learning	23

4.2.3	Generative Adversarial Networks	23
4.3	Transfer Learning	24
4.4	Semi-supervised Learning	24
4.5	Hierarchical Reinforcement Learning	24
4.6	Learning to Learn	25
5	Applications	25
5.1	Games	26
5.1.1	Perfect Information Board Games	26
5.1.2	Imperfect Information Board Games	28
5.1.3	Video Games	28
5.2	Robotics	29
5.2.1	Guided Policy Search	29
5.2.2	Learn to Navigate	29
5.3	Natural Language Processing	29
5.3.1	Dialogue Systems	30
5.3.2	Machine Translation	31
5.3.3	Text Generation	31
5.4	Computer Vision	32
5.5	Neural Architecture Design	32
5.6	Business Management	33
5.7	Finance	33
5.8	Healthcare	33
5.9	Industry 4.0	34
5.10	Smart Grid	34
5.11	Intelligent Transportation Systems	34
5.12	Computer Systems	35
6	More Topics	35
7	Resources	36
7.1	Books	37
7.2	More Books	37
7.3	Surveys and Reports	37
7.4	Courses	37
7.5	Tutorials	38
7.6	Conferences, Journals and Workshops	38
7.7	Blogs	39
7.8	Testbeds	39

7.9	Algorithm Implementations	40
8	Brief Summary	41
9	Discussions	43

1 INTRODUCTION

Reinforcement learning (RL) is about an agent interacting with the environment, learning an optimal policy, by trial and error, for sequential decision making problems in a wide range of fields in both natural and social sciences, and engineering (Sutton and Barto, 1998; 2017; Bertsekas and Tsitsiklis, 1996; Bertsekas, 2012; Szepesvári, 2010; Powell, 2011).

The integration of reinforcement learning and neural networks has a long history (Sutton and Barto, 2017; Bertsekas and Tsitsiklis, 1996; Schmidhuber, 2015). With recent exciting achievements of deep learning (LeCun et al., 2015; Goodfellow et al., 2016), benefiting from big data, powerful computation, new algorithmic techniques, mature software packages and architectures, and strong financial support, we have been witnessing the renaissance of reinforcement learning (Krakovsky, 2016), especially, the combination of deep neural networks and reinforcement learning, i.e., deep reinforcement learning (deep RL).

Deep learning, or deep neural networks, has been prevailing in reinforcement learning in the last several years, in games, robotics, natural language processing problem, etc. We have been witnessing breakthroughs, like deep Q-network (Mnih et al., 2015), AlphaGo (Silver et al., 2016a) and differentiable neural computer (Graves et al., 2016); and novel architectures and applications, like asynchronous methods (Mnih et al., 2016), dueling network architectures (Wang et al., 2016c), value iteration networks (Tamar et al., 2016), unsupervised reinforcement and auxiliary learning (Jaderberg et al., 2017; Mirowski et al., 2017), neural architecture design (Zoph and Le, 2017), dual learning for machine translation (He et al., 2016a), spoken dialogue systems (Su et al., 2016b), information extraction (Narasimhan et al., 2016), guided policy search (Levine et al., 2016a), and generative adversarial imitation learning (Ho and Ermon, 2016), etc.

Why has deep learning been helping reinforcement learning make so many and so enormous achievements? Representation learning with deep learning enables automatic feature engineering and end-to-end learning through gradient descent, so that reliance on domain knowledge is significantly reduced or even removed. Feature engineering used to be done manually and is usually time-consuming, over-specified, and incomplete. Deep, distributed representations exploit the hierarchical composition of factors in data to combat the exponential challenges of the curse of dimensionality. Generality, expressiveness and flexibility of deep neural networks make some tasks easier or possible, e.g., in the breakthroughs and novel architectures and applications discussed above.

Deep learning and reinforcement learning, being selected as one of the MIT Technology Review 10 Breakthrough Technologies in 2013 and 2017 respectively, will play their crucial role in achieving artificial general intelligence. David Silver, the major contributor of AlphaGo (Silver et al., 2016a), even made a formula: artificial intelligence = reinforcement learning + deep learning (Silver, 2016). We invent a sentence: **Deep reinforcement learning is artificial intelligence.**

The outline of this overview follows. First we discuss background of machine learning, deep learning and reinforcement learning in Section 2. Next we discuss core RL elements, including value function, policy in Section 3.2, reward in Section 3.3, model in Section 3.4, planning in Section 3.5, and exploration in Section 3.6. Then we discuss important mechanisms for RL, including attention and memory in Section 4.1, unsupervised learning in Section 4.2, transfer learning in Section 4.3, semi-supervised learning in Section 4.4, hierarchical RL in Section 4.5, and, learning to learn in Section 4.6. After that, we discuss various RL applications, including games in Section 5.1, robotics in Section 5.2, natural language processing in Section 5.3, computer vision in Section 5.4, neural architecture design in Section 5.5, business management in Section 5.6, finance in Section 5.7, healthcare in Section 5.8, Industry 4.0 in Section 5.9, smart grid in Section 5.10, intelligent transportation systems in Section 5.11, and computer systems in Section 5.12. We present a list of topics not reviewed yet in Section 6, a brief summary in Section 8, and close with discussions in Section 9.

In particular, in Section 7, we list a collection of RL resources including books, surveys, reports, online courses, tutorials, conferences, journals and workshops, blogs, and open sources. If picking a single RL resource, it is Sutton and Barto’s RL book (Sutton and Barto, 2017), 2nd edition in preparation. It covers RL fundamentals and reflects new progress, e.g., in deep Q-network, AlphaGo, policy gradient methods, as well as in psychology and neuroscience. A single pick for deep learning is Goodfellow et al. (2016). Bishop (2011), Hastie et al. (2009), and Murphy (2012) are popular machine learning textbooks; James et al. (2013) gives an introduction to machine learning,

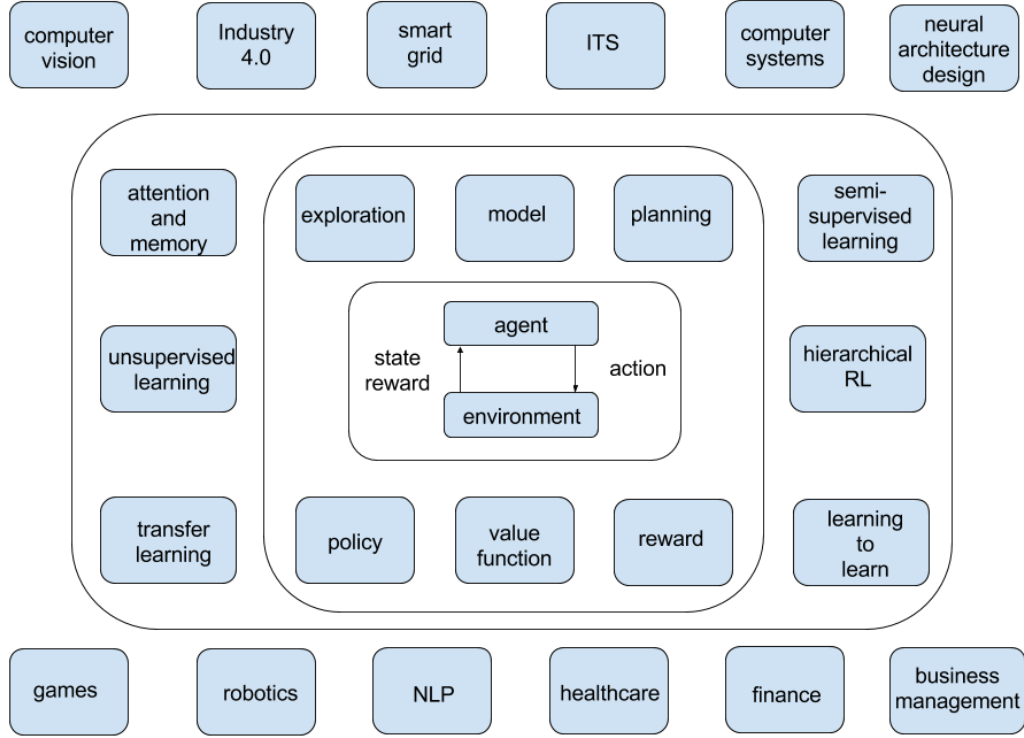


Figure 1: Conceptual Organization of the Overview

and Provost and Fawcett (2013) and Kuhn and Johnson (2013) discuss practical issues in machine learning applications.

Figure 1 illustrates the conceptual organization of the overview. The agent-environment interaction sits in the center, around which are core elements: value function, policy, reward, model, planning, and exploration. Next come important mechanisms: attention and memory, unsupervised learning, transfer learning, semi-supervised learning, hierarchical RL, and learning to learn. Then come various applications: games, robotics, NLP (natural language processing), computer vision, neural architecture design, personalized web services, healthcare, finance, Industry 4.0, smart grid, ITS (intelligent transportation systems), and computer systems.

The main readers of this overview would be those who want to get more familiar with deep reinforcement learning. We endeavour to provide as much relevant information as possible. For reinforcement learning experts, as well as new comers, we hope this overview would be helpful as a reference. In this overview, we mainly focus on contemporary work in recent couple of years, by no means complete, and make slight effort for discussions of historical context, for which the best material to consult is Sutton and Barto (2017).

In this version, we endeavour to provide a wide coverage of fundamental and contemporary RL issues, about core elements, important mechanisms, and applications. In the future, besides further refinements for the width, we will also improve the depth by conducting deeper analysis of the issues involved and the papers discussed. Comments and criticisms are welcome.

2 BACKGROUND

In this section, we briefly introduce concepts and fundamentals in machine learning, deep learning (Goodfellow et al., 2016) and reinforcement learning (Sutton and Barto, 2017). We do not give detailed background introduction for machine learning and deep learning. Instead, we recommend the following recent Nature/Science survey papers: Jordan and Mitchell (2015) for machine learn-

ing, and LeCun et al. (2015) for deep learning. We cover some RL basics. However, we recommend the textbook, Sutton and Barto (2017), and the recent Nature survey paper, Littman (2015), for reinforcement learning. We also collect relevant resources in Section 7.

2.1 MACHINE LEARNING

Machine learning is about learning from data and making predictions and/or decisions.

Usually we categorize machine learning as supervised, unsupervised, and reinforcement learning. In supervised learning, there are labeled data; in unsupervised learning, there are no labeled data; and in reinforcement learning, there are evaluative feedbacks, but no supervised signals. Classification and regression are two types of supervised learning problems, with categorical and numerical outputs respectively.

Unsupervised learning attempts to extract information from data without labels, e.g., clustering and density estimation. Representation learning is a classical type of unsupervised learning. However, training feedforward networks or convolutional neural networks with supervised learning is a kind of representation learning. Representation learning finds a representation to preserve as much information about the original data as possible, at the same time, to keep the representation simpler or more accessible than the original data, with low-dimensional, sparse, and independent representations.

Deep learning, or deep neural networks, is a particular machine learning scheme, usually for supervised or unsupervised learning, and can be integrated with reinforcement learning, usually as a function approximator. Supervised and unsupervised learning are usually one-shot, myopic, considering instant reward; while reinforcement learning is sequential, far-sighted, considering long-term accumulative reward.

Machine learning is based on probability theory and statistics (Hastie et al., 2009) and optimization (Boyd and Vandenberghe, 2004), is the basis for big data, data science, data mining, information retrieval, etc, and becomes a critical ingredient for computer vision, natural language processing, robotics, etc. Reinforcement learning is kin to optimal control (Bertsekas, 2012), and operations research and management (Powell, 2011), and is also related to psychology and neuroscience (Sutton and Barto, 2017). Machine learning is a subset of artificial intelligence (AI), and is evolving to be critical for all fields of AI.

A machine learning algorithm is composed of a dataset, a cost/loss function, an optimization procedure, and a model (Goodfellow et al., 2016). A dataset is divided into non-overlapping training, validation, and testing subsets. A cost/loss function measures the model performance, e.g., with respect to accuracy, like mean square error in regression and classification error rate. Training error measures the error on the training data, minimizing which is an optimization problem. Generalization error, or test error, measures the error on new input data, which differentiates machine learning from optimization. A machine learning algorithm tries to make the training error, and the gap between training error and testing error small. A model is under-fitting if it can not achieve a low training error; a model is over-fitting if the gap between training error and test error is large.

A model's capacity measures the range of functions it can fit. VC dimension measures the capacity of a binary classifier. Occam's Razor states that, with the same expressiveness, simple models are preferred. Training error and generalization error versus model capacity usually form a U-shape relationship. We find the optimal capacity to achieve low training error and small gap between training error and generalization error. Bias measures the expected deviation of the estimator from the true value; while variance measures the deviation of the estimator from the expected value, or variance of the estimator. As model capacity increases, bias tends to decrease, while variance tends to increase, yielding another U-shape relationship between generalization error versus model capacity. We try to find the optimal capacity point, of which under-fitting occurs on the left and over-fitting occurs on the right. Regularization add a penalty term to the cost function, to reduce the generalization error, but not training error. No free lunch theorem states that there is no universally best model, or best regularizer. An implication is that deep learning may not be the best model for some problems. There are model parameters, and hyperparameters for model capacity and regularization. Cross-validation is used to tune hyperparameters, to strike a balance between bias and variance, and to select the optimal model.

Maximum likelihood estimation (MLE) is a common approach to derive good estimation of parameters. For issues like numerical underflow, the product in MLE is converted to summation to obtain negative log-likelihood (NLL). MLE is equivalent to minimizing KL divergence, the dissimilarity between the empirical distribution defined by the training data and the model distribution. Minimizing KL divergence between two distributions corresponds to minimizing the cross-entropy between the distributions. In short, maximization of likelihood becomes minimization of the negative log-likelihood (NLL), or equivalently, minimization of cross entropy.

Gradient descent is a common approach to solve optimization problems. Stochastic gradient descent extends gradient descent by working with a single sample each time, and usually with minibatches.

Importance sampling is a technique to estimate properties of a particular distribution, by samples from a different distribution, to lower the variance of the estimation, or when sampling from the distribution of interest is difficult.

Frequentist statistics estimates a single value, and characterizes variance by confidence interval; Bayesian statistics considers the distribution of an estimate when making predictions and decisions.

2.2 DEEP LEARNING

Deep learning is in contrast to "shallow" learning. For many machine learning algorithms, e.g., linear regression, logistic regression, support vector machines (SVMs), decision trees, and boosting, we have input layer and output layer, and the inputs may be transformed with manual feature engineering before training. In deep learning, between input and output layers, we have one or more hidden layers. At each layer except input layer, we compute the input to each unit, as the weighted sum of units from the previous layer; then we usually use nonlinear transformation, or activation function, such as logistic, tanh, or more popular recently, rectified linear unit (ReLU), to apply to the input of a unit, to obtain a new representation of the input from previous layer. We have weights on links between units from layer to layer. After computations flow forward from input to output, at output layer and each hidden layer, we can compute error derivatives backward, and backpropagate gradients towards the input layer, so that weights can be updated to optimize some loss function.

A feedforward deep neural network or multilayer perceptron (MLP) is to map a set of input values to output values with a mathematical function formed by composing many simpler functions at each layer. A convolutional neural network (CNN) is a feedforward deep neural network, with convolutional layers, pooling layers and fully connected layers. CNNs are designed to process data with multiple arrays, e.g., colour image, language, audio spectrogram, and video, benefit from the properties of such signals: local connections, shared weights, pooling and the use of many layers, and are inspired by simple cells and complex cells in visual neuroscience (LeCun et al., 2015). A recurrent neural network (RNN) is often used to process sequential inputs like speech and language, element by element, with hidden units to store history of past elements. A RNN can be seen as a multilayer neural network with all layers sharing the same weights, when being unfolded in time of forward computation. It is hard for RNN to store information for very long time and the gradient may vanish. Long short term memory networks (LSTM) and gated recurrent unit (GRU) were proposed to address such issues, with gating mechanisms to manipulate information through recurrent cells. Gradient backpropagation or its variants can be used for training all deep neural networks mentioned above.

Dropout is a regularization strategy to train an ensemble of sub-networks by removing non-output units randomly from the original network. Batch normalization performs the normalization for each training mini-batch, to accelerate training by reducing internal covariate shift, i.e., the change of parameters of previous layers will change each layer's inputs distribution.

Deep neural networks learn representations automatically from raw inputs to recover the compositional hierarchies in many natural signals, i.e., higher-level features are composed of lower-level ones, e.g., in images, the hierarchy of objects, parts, motifs, and local combinations of edges. Distributed representation is a central idea in deep learning, which implies that many features may represent each input, and each feature may represent many inputs. The exponential advantages of deep, distributed representations combat the exponential challenges of the curse of dimensionality. The notion of end-to-end training refers to that a learning model uses raw inputs without manual feature engineering to generate outputs, e.g., AlexNet (Krizhevsky et al., 2012) with raw pixels for

image classification, Seq2Seq (Sutskever et al., 2014) with raw sentences for machine translation, and DQN (Mnih et al., 2015) with raw pixels and score to play games.

2.3 REINFORCEMENT LEARNING

We provide background of reinforcement learning briefly in this section. After setting up the RL problem, we discuss value function, temporal difference learning, function approximation, policy optimization, deep RL, RL parlance, and close this section with a brief summary. To have a good understanding of deep reinforcement learning, it is essential to have a good understanding of reinforcement learning first.

2.3.1 PROBLEM SETUP

A RL agent interacts with an environment over time. At each time step t , the agent receives a state s_t in a state space \mathcal{S} and selects an action a_t from an action space \mathcal{A} , following a policy $\pi(a_t|s_t)$, which is the agent’s behavior, i.e., a mapping from state s_t to actions a_t , receives a scalar reward r_t , and transitions to the next state s_{t+1} , according to the environment dynamics, or model, for reward function $\mathcal{R}(s, a)$ and state transition probability $\mathcal{P}(s_{t+1}|s_t, a_t)$ respectively. In an episodic problem, this process continues until the agent reaches a terminal state and then it restarts. The return $R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k}$ is the discounted, accumulated reward with the discount factor $\gamma \in (0, 1]$. The agent aims to maximize the expectation of such long term return from each state. The problem is set up in discrete state and action spaces. It is not hard to extend it to continuous spaces.

2.3.2 VALUE FUNCTION

A value function is a prediction of the expected, accumulative, discounted, future reward, measuring how good each state, or state-action pair, is. The state value $v_{\pi}(s) = E[R_t|s_t = s]$ is the expected return for following policy π from state s . $v_{\pi}(s)$ decomposes into the Bellman equation: $v_{\pi}(s) = \sum_a \pi(a|s) \sum_{s', r} p(s', r|s, a) [r + \gamma v_{\pi}(s')]$. An optimal state value $v_*(s) = \max_{\pi} v_{\pi}(s)$ is the maximum state value achievable by any policy for state s . $v_*(s)$ decomposes into the Bellman equation: $v_*(s) = \sum_a \pi(a|s) \sum_{s', r} p(s', r|s, a) [r + \gamma v_*(s')]$. The action value $q_{\pi}(s, a) = E[R_t|s_t = s, a_t = a]$ is the expected return for selecting action a in state s and then following policy π . $q_{\pi}(s, a)$ decomposes into the Bellman equation: $q_{\pi}(s, a) = \sum_{s', r} p(s', r|s, a) [r + \gamma \max_{a'} q_{\pi}(s', a')]$. An optimal action value function $q_*(s, a) = \max_{\pi} q_{\pi}(s, a)$ is the maximum action value achievable by any policy for state s and action a . $q_*(s, a)$ decomposes into the Bellman equation: $q_*(s, a) = \sum_{s', r} p(s', r|s, a) [r + \gamma \max_{a'} q_*(s', a')]$. We denote an optimal policy by π^* .

2.3.3 TEMPORAL DIFFERENCE LEARNING

When a RL problem satisfies the Markov property, i.e., the future depends only on the current state and action, but not on the past, it is formulated as a Markov Decision Process (MDP), defined by the 5-tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$. When the system model is available, we use dynamic programming methods: policy evaluation to calculate value/action value function for a policy, value iteration and policy iteration for finding an optimal policy. When there is no model, we resort to RL methods. RL methods also work when the model is available. Additionally, a RL environment can be a multi-armed bandit, an MDP, a POMDP, a game, etc.

Temporal difference (TD) learning is central in RL. TD learning is usually refer to the learning methods for value function evaluation in Sutton (1988). SARSA (Sutton and Barto, 2017) and Q-learning (Watkins and Dayan, 1992) are also regarded as temporal difference learning.

TD learning (Sutton, 1988) learns value function $V(s)$ directly from experience with TD error, with bootstrapping, in a model-free, online, and fully incremental way. TD learning is a prediction problem. The update rule is $V(s) \leftarrow V(s) + \alpha [r + \gamma V(s') - V(s)]$, where α is a learning rate, and $r + \gamma V(s') - V(s)$ is called TD error. Algorithm 1 presents the pseudo code for tabular TD learning. Precisely, it is tabular TD(0) learning, where 0 indicates it is based on one-step return.

Bootstrapping, like the TD update rule, estimates state or action value based on subsequent estimates, is common in RL, like TD learning, Q learning, and actor-critic. Bootstrapping methods are usually faster to learn, and enable learning to be online and continual. Bootstrapping methods are

not instances of true gradient decent, since the target depends on the weights to be estimated. The concept of semi-gradient descent is then introduced (Sutton and Barto, 2017).

Input: the policy π to be evaluated
Output: value function V
 initialize V arbitrarily, e.g., to 0 for all states
for each episode do
 initialize state s
 for each step of episode, state s is not terminal do
 $a \leftarrow$ action given by π for s
 take action a , observe r, s'
 $V(s) \leftarrow V(s) + \alpha[r + \gamma V(s') - V(s)]$
 $s \leftarrow s'$
 end
end

Algorithm 1: TD learning, adapted from Sutton and Barto (2017)

Output: action value function Q
 initialize Q arbitrarily, e.g., to 0 for all states, set action value for terminal states as 0
for each episode do
 initialize state s
 for each step of episode, state s is not terminal do
 $a \leftarrow$ action for s derived by Q , e.g., ϵ -greedy
 take action a , observe r, s'
 $a' \leftarrow$ action for s' derived by Q , e.g., ϵ -greedy
 $Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma Q(s', a') - Q(s, a)]$
 $s \leftarrow s', a \leftarrow a'$
 end
end

Algorithm 2: SARSA, adapted from Sutton and Barto (2017)

Output: action value function Q
 initialize Q arbitrarily, e.g., to 0 for all states, set action value for terminal states as 0
for each episode do
 initialize state s
 for each step of episode, state s is not terminal do
 $a \leftarrow$ action for s derived by Q , e.g., ϵ -greedy
 take action a , observe r, s'
 $Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma \max_{a'} Q(s', a') - Q(s, a)]$
 $s \leftarrow s'$
 end
end

Algorithm 3: Q learning, adapted from Sutton and Barto (2017)

SARSA, representing state, action, reward, (next) state, (next) action, is an on-policy control method to find the optimal policy, with the update rule, $Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma Q(s', a') - Q(s, a)]$. Algorithm 2 presents the pseudo code for tabular SARSA, precisely tabular SARSA(0).

Q-learning is an off-policy control method to find the optimal policy. Q-learning learns action value function, with the update rule, $Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma \max_{a'} Q(s', a') - Q(s, a)]$. Q learning refines the policy greedily with respect to action values by the max operator. Algorithm 3 presents the pseudo code for Q learning, precisely, tabular Q(0) learning.

TD-learning, Q-learning and SARSA converge under certain conditions. From an optimal action value function, we can derive an optimal policy.

2.3.4 MULTI-STEP BOOTSTRAPPING

The above algorithms are referred to as TD(0) and Q(0), learning with one-step return. We have TD learning and Q learning variants and Monte-Carlo approach with multi-step return in the forward

view. The eligibility trace from the backward view provides an online, incremental implementation, resulting in TD(λ) and Q(λ) algorithms, where $\lambda \in [0, 1]$. TD(1) is the same as the Monte Carlo approach.

Eligibility trace is a short-term memory, usually lasting within an episode, assists the learning process, by affecting the weight vector. The weight vector is a long-term memory, lasting the whole duration of the system, determines the estimated value. Eligibility trace helps with the issues of long-delayed rewards and non-Markov tasks (Sutton and Barto, 2017).

TD(λ) unifies one-step TD prediction, TD(0), with Monte Carlo methods, TD(1), using eligibility traces and the decay parameter λ , for prediction algorithms. De Asis et al. (2017) made unification for multi-step TD control algorithms.

2.3.5 FUNCTION APPROXIMATION

We discuss the tabular cases above, where a value function or a policy is stored in a tabular form. Function approximation is a way for generalization when the state and/or action spaces are large or continuous. Function approximation aims to generalize from examples of a function to construct an approximate of the entire function; it is usually a concept in supervised learning, studied in the fields of machine learning, pattern recognition, and statistical curve fitting; function approximation in reinforcement learning usually treats each backup as a training example, and encounters new issues like nonstationarity, bootstrapping, and delayed targets (Sutton and Barto, 2017). Linear function approximation is a popular choice, partially due to its desirable theoretical properties, esp. before the work of Deep Q-Network (Mnih et al., 2015). However, the integration of reinforcement learning and neural networks dated back a long time ago (Sutton and Barto, 2017; Bertsekas and Tsitsiklis, 1996; Schmidhuber, 2015).

Algorithm 4 presents the pseudo code for TD(0) with function approximation. $\hat{v}(s, \mathbf{w})$ is the approximate value function, \mathbf{w} is the value function weight vector, $\nabla \hat{v}(s, \mathbf{w})$ is the gradient of the approximate value function with respect to the weight vector, and the weight vector is updated following the update rule, $\mathbf{w} \leftarrow \mathbf{w} + \alpha[r + \gamma \hat{v}(s', \mathbf{w}) - \hat{v}(s, \mathbf{w})] \nabla \hat{v}(s, \mathbf{w})$.

Input: the policy π to be evaluated

Input: a differentiable value function $\hat{v}(s, \mathbf{w})$, $\hat{v}(\text{terminal}, \cdot) = 0$

Output: value function $\hat{v}(s, \mathbf{w})$

initialize value function weight \mathbf{w} arbitrarily, e.g., $\mathbf{w} = 0$

```

for each episode do
  initialize state  $s$ 
  for each step of episode, state  $s$  is not terminal do
     $a \leftarrow \pi(\cdot | s)$ 
    take action  $a$ , observe  $r, s'$ 
     $\mathbf{w} \leftarrow \mathbf{w} + \alpha[r + \gamma \hat{v}(s', \mathbf{w}) - \hat{v}(s, \mathbf{w})] \nabla \hat{v}(s, \mathbf{w})$ 
     $s \leftarrow s'$ 
  end
end

```

Algorithm 4: TD(0) with function approximation, adapted from Sutton and Barto (2017)

When combining off-policy, function approximation, and bootstrapping, instability and divergence may occur (Tsitsiklis and Van Roy, 1997), which is called the deadly triad issue (Sutton and Barto, 2017). All these three elements are necessary: function approximation for scalability and generalization, bootstrapping for computational and data efficiency, and off-policy learning for freeing behaviour policy from target policy. What is the root cause for the instability? Learning or sampling are not, since dynamic programming suffers from divergence with function approximation; exploration, greedification, or control are not, since prediction alone can diverge; local minima or complex non-linear function approximation are not, since linear function approximation can produce instability (Sutton, 2016). It is unclear what is the root cause for instability – each single factor mentioned above is not – there are still many open problems in off-policy learning (Sutton and Barto, 2017).

Table 1 presents various algorithms that tackle various issues (Sutton, 2016). Deep RL algorithms like Deep Q-Network (Mnih et al., 2015) and A3C (Mnih et al., 2016) are not presented here, since they do not have theoretical guarantee, although they achieve stunning performance empirically.

Before explaining Table 1, we introduce some background definitions. Recall that Bellman equation for value function is $v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)[r + \gamma v_\pi(s')]$. Bellman operator is defined as $(B_\pi v)(s) \doteq \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)[r + \gamma v_\pi(s')]$. TD fix point is then $v_\pi = B_\pi v_\pi$. Bellman error for the function approximation case is then $\sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)[r + \gamma \hat{v}^\pi(s', \mathbf{w})] - \hat{v}^\pi(s, \mathbf{w})$, the right side of Bellman equation with function approximation minus the left side. It can be written as $B_\pi v_\mathbf{w} - v_\mathbf{w}$. Bellman error is the expectation of the TD error.

ADP algorithms refer to dynamic programming algorithms like policy evaluation, policy iteration, and value iteration, with function approximation. Least square temporal difference (LSTD) (Bradtke and Barto, 1996) computes TD fix-point directly in batch mode. LSTD is data efficient, yet with squared time complexity. LSPE (Nedić and Bertsekas, 2003) extended LSTD. Fitted-Q algorithms (Ernst et al., 2005; Riedmiller, 2005) learn action values in batch mode. Residual gradient algorithms (Baird, 1995) minimize Bellman error. Gradient-TD (Sutton et al., 2009a;b; Mahmood et al., 2014) methods are true gradient algorithms, perform SGD in the projected Bellman error (PBE), converge robustly under off-policy training and non-linear function approximation. Emphatic-TD (Sutton et al., 2016) emphasizes some updates and de-emphasizes others by reweighting, improving computational efficiency, yet being a semi-gradient method. See Sutton and Barto (2017) for more details. Du et al. (2017) proposed variance reduction techniques for policy evaluation to achieve fast convergence.

		algorithm					
		TD(λ) SARSA(λ)	ADP	LSTD(λ) LSPE(λ)	Fitted-Q	Residual Gradient	GTD(λ) GQ(λ)
issue	linear computation	✓	✓			✓	✓
	nonlinear convergent				✓	✓	✓
	off-policy convergent			✓		✓	✓
	model-free, online	✓		✓		✓	✓
	converges to PBE = 0	✓	✓	✓	✓		✓

Table 1: RL Issues vs. Algorithms

2.3.6 POLICY OPTIMIZATION

In contrast to value-based methods like TD learning and Q-learning, policy-based methods optimize the policy $\pi(a|s; \theta)$ (with function approximation) directly, and update the parameters θ by gradient ascent on $E[R_t]$. REINFORCE (Williams, 1992) is a policy gradient method, updating θ in the direction of $\nabla_\theta \log \pi(a_t|s_t; \theta) R_t$. Usually a baseline $b_t(s_t)$ is subtracted from the return to reduce the variance of gradient estimate, yet keeping its unbiasedness, to yield the gradient direction $\nabla_\theta \log \pi(a_t|s_t; \theta) (R_t - b_t(s_t))$. Using $V(s_t)$ as the baseline $b_t(s_t)$, we have the advantage function $A(a_t, s_t) = Q(a_t, s_t) - V(s_t)$, since R_t is an estimate of $Q(a_t, s_t)$. Algorithm 5 presents the pseudo code for REINFORCE algorithm in the episodic case.

In actor-critic algorithms, the critic updates action-value function parameters, and the actor updates policy parameters, in the direction suggested by the critic. Algorithm 6 presents the pseudo code for one-step actor-critic algorithm in the episodic case.

2.3.7 DEEP REINFORCEMENT LEARNING

We obtain deep reinforcement learning (deep RL) methods when we use deep neural networks to approximate any of the following component of reinforcement learning: value function, $\hat{v}(s; \theta)$ or $\hat{q}(s, a; \theta)$, policy $\pi(a|s; \theta)$, and model (state transition function and reward function). Here, the parameters θ are the weights in deep neural networks. When we use "shallow" models, like linear function, decision trees, tile coding and so on as the function approximator, we obtain "shallow" RL, and the parameters θ are the weight parameters in these models. Note, a shallow model, e.g., decision trees, may be non-linear. The distinct difference between deep RL and "shallow" RL is

Input: policy $\pi(a|s, \theta)$, $\hat{v}(s, w)$
Parameters: step sizes, $\alpha > 0, \beta > 0$
Output: policy $\pi(a|s, \theta)$
initialize policy parameter θ and state-value weights w
for true do
 generate an episode $s_0, a_0, r_1, \dots, s_{T-1}, a_{T-1}, r_T$, following $\pi(\cdot|\cdot, \theta)$
 for each step t of episode $0, \dots, T-1$ do
 $G_t \leftarrow$ return from step t
 $\delta \leftarrow G_t - \hat{v}(s_t, w)$
 $w \leftarrow w + \beta \delta \nabla_w \hat{v}(s_t, w)$
 $\theta \leftarrow \theta + \alpha \gamma^t \delta \nabla_\theta \log \pi(a_t|s_t, \theta)$
 end
end
Algorithm 5: REINFORCE with baseline (episodic), adapted from Sutton and Barto (2017)

Input: policy $\pi(a|s, \theta)$, $\hat{v}(s, w)$
Parameters: step sizes, $\alpha > 0, \beta > 0$
Output: policy $\pi(a|s, \theta)$
initialize policy parameter θ and state-value weights w
for true do
 initialize s , the first state of the episode
 $I \leftarrow 1$
 for s is not terminal do
 $a \sim \pi(\cdot|s, \theta)$
 take action a , observe s', r
 $\delta \leftarrow r + \gamma \hat{v}(s', w) - \hat{v}(s, w)$ (if s' is terminal, $\hat{v}(s', w) \doteq 0$)
 $w \leftarrow w + \beta \delta \nabla_w \hat{v}(s_t, w)$
 $\theta \leftarrow \theta + \alpha I \delta \nabla_\theta \log \pi(a_t|s_t, \theta)$
 $I \leftarrow \gamma I$
 $s \leftarrow s'$
 end
end
Algorithm 6: Actor-Critic (episodic), adapted from Sutton and Barto (2017)

what function approximator is used. This is similar to the difference between deep learning and "shallow" learning. We usually utilize stochastic gradient descent to update weight parameters in deep RL. When off-policy, function approximation, in particular, non-linear function approximation, and bootstrapping are combined together, instability and divergence may occur (Tsitsiklis and Van Roy, 1997). However, recent work like Deep Q-Network (Mnih et al., 2015) and AlphaGo (Silver et al., 2016a) stabilized the learning and achieved outstanding results.

2.3.8 RL PARLANCE

We explain some terms in RL parlance.

The prediction problem, or policy evaluation, is to compute the state or action value function for a policy. The control problem is to find the optimal policy. Planning constructs a value function or a policy with a model.

On-policy methods evaluate or improve the behavioural policy, e.g., SARSA fits the action-value function to the current policy, i.e., SARSA evaluates the policy based on samples from the same policy, then refines the policy greedily with respect to action values. In off-policy methods, an agent learns an optimal value function/policy, maybe following an unrelated behavioural policy, e.g., Q-learning attempts to find action values for the optimal policy directly, not necessarily fitting to the policy generating the data, i.e., the policy Q-learning obtains is usually different from the policy that generates the samples. The notion of on-policy and off-policy can be understood as same-policy and different-policy.

The exploration-exploitation dilemma is about the agent needs to exploit the currently best action to maximize rewards greedily, yet it has to explore the environment to find better actions, when the policy is not optimal yet, or the system is non-stationary.

In model-free methods, the agent learns with trial-and-error from experience explicitly; the model (state transition function) is not known or learned from experience. RL methods that use models are model-based methods.

In online mode, training algorithms are executed on data acquired in sequence. In offline mode, or batch mode, models are trained on the entire data set.

With bootstrapping, an estimate of state or action value is updated from subsequent estimates.

2.3.9 BRIEF SUMMARY

A RL problem is formulated as an MDP when the observation about the environment satisfies the Markov property. An MDP is defined by the 5-tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$. A central concept in RL is value function. Bellman equations are cornerstone for developing RL algorithms. Temporal difference learning algorithms are fundamental for evaluating/predicting value functions. Control algorithms find optimal policies. Reinforcement learning algorithms may be based on value function and/or policy, model-free or model-based, on-policy or off-policy, with function approximation or not, with sample backups (TD and Monte Carlo) or full backups (dynamic programming and exhaustive search), and about the depth of backups, either one-step return (TD(0) and dynamic programming) or multi-step return (TD(λ), Monte Carlo, and exhaustive search). When combining off-policy, function approximation, and bootstrapping, we face instability and divergence (Tsitsiklis and Van Roy, 1997), the deadly triad issue (Sutton and Barto, 2017). Theoretical guarantee has been established for linear function approximation, e.g., Gradient-TD (Sutton et al., 2009a;b; Mahmood et al., 2014), Emphatic-TD (Sutton et al., 2016) and Du et al. (2017). With non-linear function approximation, in particular deep learning, algorithms like Deep Q-Network (Mnih et al., 2015) and AlphaGo (Silver et al., 2016a) stabilized the learning and achieved stunning results, which is the focus of this overview.

3 CORE ELEMENTS

A RL agent executes a sequence of actions and observe states and rewards, with major components of value function, policy and model. A RL problem may be formulated as a prediction, control or planning problem, and solution methods may be model-free or model-based, with value function and/or policy. Exploration-exploitation is a fundamental tradeoff in RL.

In this section, we discuss core RL elements: value function in Section 3.1, policy in Section 3.2, reward in Section 3.3, model in Section 3.4, planning in Section 3.5, and exploration in Section 3.6.

3.1 VALUE FUNCTION

Value function is a fundamental concept in reinforcement learning, and temporal difference (TD) learning (Sutton, 1988) and its extension, Q-learning (Watkins and Dayan, 1992), are classical algorithms for learning state and action value functions respectively. In the following, we focus on Deep Q-Network (Mnih et al., 2015), a recent breakthrough, and its extensions.

3.1.1 DEEP Q-NETWORK (DQN)

Mnih et al. (2015) introduced Deep Q-Network (DQN) and ignited the field of deep RL. We present DQN pseudo code in Algorithm 7.

Before DQN, it is well known that RL is unstable or even divergent when action value function is approximated with a nonlinear function like neural networks. DQN made several important contributions: 1) stabilize the training of action value function approximation with deep neural networks (CNN) using experience replay (Lin, 1992) and target network; 2) designing an end-to-end RL approach, with only the pixels and the game score as inputs, so that only minimal domain knowledge is required; 3) training a flexible network with the same algorithm, network architecture and hyper-

Input: the pixels and the game score
Output: Q action value function (from which we obtain policy and select action)
Initialize replay memory D
Initialize action-value function Q with random weight θ
Initialize target action-value function \hat{Q} with weights $\theta^- = \theta$
for $episode = 1$ **to** M **do**
 Initialize sequence $s_1 = \{x_1\}$ and preprocessed sequence $\phi_1 = \phi(s_1)$
 for $t = 1$ **to** T **do**
 Following ϵ -greedy policy, select $a_t = \begin{cases} \text{a random action} & \text{with probability } \epsilon \\ \arg \max_a Q(\phi(s_t), a; \theta) & \text{otherwise} \end{cases}$
 Execute action a_t in emulator and observe reward r_t and image x_{t+1}
 Set $s_{t+1} = s_t, a_t, x_{t+1}$ and preprocess $\phi_{t+1} = \phi(s_{t+1})$
 Store transition $(\phi_t, a_t, r_t, \phi_{t+1})$ in D
 // experience replay
 Sample random minibatch of transitions $(\phi_j, a_j, r_j, \phi_{j+1})$ from D
 Set $y_j = \begin{cases} r_j & \text{if episode terminates at step } j+1 \\ r_j + \gamma \max_{a'} \hat{Q}(\phi_{j+1}, a'; \theta^-) & \text{otherwise} \end{cases}$
 Perform a gradient descent step on $(y_j - Q(\phi_j, a_j; \theta))^2$ w.r.t. the network parameter θ
 // periodic update of target network
 Every C steps reset $\hat{Q} = Q$, i.e., set $\theta^- = \theta$
 end
end

Algorithm 7: Deep Q-Network (DQN), adapted from Mnih et al. (2015)

parameters to perform well on many different tasks, i.e., 49 Atari games (Bellemare et al., 2013), and outperforming previous algorithms and performing comparably to a human professional tester.

See Chapter 16 in Sutton and Barto (2017) for a detailed and intuitive description of Deep Q-Network. See Deepmind’s description of DQN at <https://deepmind.com/research/dqn/>.

3.1.2 DOUBLE DQN

van Hasselt et al. (2016a) proposed Double DQN (D-DQN) to tackle the over-estimate problem in Q-learning. In standard Q-learning, as well as in DQN, the parameters are updated as follows:

$$\theta_{t+1} = \theta_t + \alpha(y_t^Q - Q(s_t, a_t; \theta_t)) \nabla_{\theta_t} Q(s_t, a_t; \theta_t),$$

where

$$y_t^Q = r_{t+1} + \gamma \max_a Q(s_{t+1}, a; \theta_t),$$

so that the max operator uses the same values to both select and evaluate an action. As a consequence, it is more likely to select over-estimated values, and results in over-optimistic value estimates. van Hasselt et al. (2016a) proposed to evaluate the greedy policy according to the online network, but to use the target network to estimate its value. This can be achieved with a minor change to the DQN algorithm, replacing y_t^Q with

$$y_t^{D-DQN} = r_{t+1} + \gamma Q(s_{t+1}, \arg \max_a Q(s_{t+1}, a_t; \theta_t); \theta_t^-),$$

where θ_t is the parameter for online network and θ_t^- is the parameter for target network. For reference, y_t^Q can be written as

$$y_t^Q = r_{t+1} + \gamma Q(s_{t+1}, \arg \max_a Q(s_{t+1}, a_t; \theta_t); \theta_t).$$

D-DQN found better policies than DQN on Atari games.

3.1.3 PRIORITIZED EXPERIENCE REPLAY

In DQN, experience transitions are uniformly sampled from the replay memory, regardless of the significance of experiences. Schaul et al. (2016) proposed to prioritize experience replay, so that

important experience transitions can be replayed more frequently, to learn more efficiently. The importance of experience transitions are measured by TD errors. The authors designed a stochastic prioritization based on the TD errors, using importance sampling to avoid the bias in the update distribution. The authors used prioritized experience replay in DQN and D-DQN, and improved their performance on Atari games.

3.1.4 DUELING ARCHITECTURE

Wang et al. (2016c) proposed the dueling network architecture to estimate state value function $V(s)$ and associated advantage function $A(s, a)$, and then combine them to estimate action value function $Q(s, a)$, to converge faster than Q-learning. In DQN, a CNN layer is followed by a fully connected (FC) layer. In dueling architecture, a CNN layer is followed by two streams of FC layers, to estimate value function and advantage function separately; then the two streams are combined to estimate action value function. Usually we use the following to combine $V(s)$ and $A(s, a)$ to obtain $Q(s, a)$,

$$Q(s, a; \theta, \alpha, \beta) = V(s; \theta, \beta) + (A(s, a; \theta, \alpha) - \max_{a'} A(s, a'; \theta, \alpha))$$

where α and β are parameters of the two streams of FC layers. Wang et al. (2016c) proposed to replace max operator with average as following for better stability,

$$Q(s, a; \theta, \alpha, \beta) = V(s; \theta, \beta) + (A(s, a; \theta, \alpha) - \frac{a}{|A|} A(s, a'; \theta, \alpha))$$

Dueling architecture implemented with D-DQN and prioritized experience replay improved previous work, DQN and D-DQN with prioritized experience replay, on Atari games.

3.1.5 MORE DQN EXTENSIONS

DQN has been receiving much attention. We list several extensions/improvements here.

- Anschel et al. (2017) proposed to reduce variability and instability by an average of previous Q-values estimates.
- He et al. (2017a) proposed to accelerate DQN by optimality tightening, a constrained optimization approach, to propagate reward faster, and to improve accuracy over DQN.
- Liang et al. (2016) attempted to understand the success of DQN and reproduced results with shallow RL.
- O'Donoghue et al. (2017) proposed policy gradient and Q-learning (PGQ), as discussed in Section 3.2.3.
- Oh et al. (2015) proposed spatio-temporal video prediction conditioned on actions and previous video frames with deep neural networks in Atari games.
- Osband et al. (2016) designed better exploration strategy to improve DQN.

3.2 POLICY

A policy maps state to action, and policy optimization is to find an optimal mapping. We discuss actor-critic (Mnih et al., 2016). Then we discuss policy gradient, including deterministic policy gradient (Silver et al., 2014; Lillicrap et al., 2016), trust region policy optimization (Schulman et al., 2015), and, benchmark results (Duan et al., 2016). Next we discuss the combination of policy gradient and off-policy RL (O'Donoghue et al., 2017; Nachum et al., 2017; Gu et al., 2017).

See Retrace algorithm (Munos et al., 2016), a safe and efficient return-based off-policy control algorithm, and its actor-critic extension, Reactor (Gruslys et al., 2017), for Retrace-actor. See distributed proximal policy optimization (Heess et al., 2017).

3.2.1 ACTOR-CRITIC

An actor-critic algorithm learns both a policy and a state-value function, and the value function is used for bootstrapping, i.e., updating a state from subsequent estimates, to reduce variance and accelerate learning (Sutton and Barto, 2017). In the following, we focus on asynchronous advantage

Global shared parameter vectors θ and θ_v , thread-specific parameter vectors θ' and θ'_v
Global shared counter $T = 0, T_{max}$
Initialize step counter $t \leftarrow 1$
for $T \leq T_{max}$ **do**
 Reset gradients, $d\theta \leftarrow 0$ and $d\theta_v \leftarrow 0$
 Synchronize thread-specific parameters $\theta' = \theta$ and $\theta'_v = \theta_v$
 Set $t_{start} = t$, get state s_t
 for s_t not terminal and $t - t_{start} \leq t_{max}$ **do**
 Take a_t according to policy $\pi(a_t|s_t; \theta')$
 Receive reward r_t and new state s_{t+1}
 $t \leftarrow t + 1, T \leftarrow T + 1$
 end
 $R = \begin{cases} 0 & \text{for terminal } s_t \\ V(s_t, \theta'_v) & \text{otherwise} \end{cases}$
 for $i \in \{t - 1, \dots, t_{start}\}$ **do**
 $R \leftarrow r_i + \gamma R$
 accumulate gradients wrt θ' : $d\theta \leftarrow d\theta + \nabla_{\theta'} \log \pi(a_i|s_i; \theta')(R - V(s_i; \theta'_v))$
 accumulate gradients wrt θ'_v : $d\theta_v \leftarrow d\theta_v + \nabla_{\theta'_v} (R - V(s_i; \theta'_v))^2$
 end
 Update asynchronously θ using $d\theta$, and θ_v using $d\theta_v$
end

Algorithm 8: A3C, each actor-learner thread, based on Mnih et al. (2016)

actor-critic (A3C) (Mnih et al., 2016). Mnih et al. (2016) also discussed asynchronous one-step SARSA, one-step Q-learning and n-step Q-learning.

In A3C, parallel actors employ different exploration policies to stabilize training, so that experience replay is not utilized. Different from most deep learning algorithms, asynchronous methods can run on a single multi-core CPU. For Atari games, A3C ran much faster yet performed better than or comparably with DQN, Gorila, D-DQN, Dueling D-DQN, and Prioritized D-DQN. A3C also succeeded on continuous motor control problems: TORCS car racing games and MuJoCo physics manipulation and locomotion, and Labyrinth, a navigating task in random 3D mazes using visual inputs, in which an agent will face a new maze in each new episode, so that it needs to learn a general strategy to explore random mazes.

We present pseudo code for asynchronous advantage actor-critic for each actor-learner thread in Algorithm 8. A3C maintains a policy $\pi(a_t|s_t; \theta)$ and an estimate of the value function $V(s_t; \theta_v)$, being updated with n -step returns in the forward view, after every t_{max} actions or reaching a terminal state, similar to using minibatches. The gradient update can be seen as $\nabla_{\theta'} \log \pi(a_t|s_t; \theta') A(s_t, a_t; \theta, \theta_v)$, where $A(s_t, a_t; \theta, \theta_v) = \sum_{i=0}^{k-1} \gamma^i r_{t+i} + \gamma^k V(s_{t+k}; \theta_v) - V(s_t; \theta_v)$ is an estimate of the advantage function, with k upbounded by t_{max} .

Wang et al. (2017b) proposed a stable and sample efficient actor-critic deep RL model using experience replay, with truncated importance sampling, stochastic dueling network (Wang et al., 2016c) as discussed in Section 3.1.4, and trust region policy optimization (Schulman et al., 2015) as discussed in Section 3.2.2. Babaeizadeh et al. (2017) proposed a hybrid CPU/GPU implementation of A3C.

3.2.2 POLICY GRADIENT

REINFORCE (Williams, 1992; Sutton et al., 2000) is a popular policy gradient method. Relatively speaking, Q-learning as discussed in Section 3.1 is sample efficient, while policy gradient is stable.

DETERMINISTIC POLICY GRADIENT

Policies are usually stochastic. However, Silver et al. (2014) and Lillicrap et al. (2016) proposed deterministic policy gradient (DPG) for efficient estimation of policy gradients.

Silver et al. (2014) introduced the deterministic policy gradient (DPG) algorithm for RL problems with continuous action spaces. The deterministic policy gradient is the expected gradient of the

action-value function, which integrates over the state space; whereas in the stochastic case, the policy gradient integrates over both state and action spaces. Consequently, the deterministic policy gradient can be estimated more efficiently than the stochastic policy gradient. The authors introduced an off-policy actor-critic algorithm to learn a deterministic target policy from an exploratory behaviour policy, and to ensure unbiased policy gradient with the compatible function approximation for deterministic policy gradients. Empirical results showed its superior to stochastic policy gradients, in particular in high dimensional tasks, on several problems: a high-dimensional bandit; standard benchmark RL tasks of mountain car and pendulum and 2D puddle world with low dimensional action spaces; and controlling an octopus arm with a high-dimensional action space. The experiments were conducted with tile-coding and linear function approximators.

Lillicrap et al. (2016) proposed an actor-critic, model-free, deep deterministic policy gradient (DDPG) algorithm in continuous action spaces, by extending DQN (Mnih et al., 2015) and DPG (Silver et al., 2014). With actor-critic as in DPG, DDPG avoids the optimization of action at every time step to obtain a greedy policy as in Q-learning, which will make it infeasible in complex action spaces with large, unconstrained function approximators like deep neural networks. To make the learning stable and robust, similar to DQN, DDPG deploys experience replay and an idea similar to target network, "soft" target, which, rather than copying the weights directly as in DQN, updates the soft target network weights θ' slowly to track the learned networks weights θ : $\theta' \leftarrow \tau\theta + (1 - \tau)\theta'$, with $\tau \ll 1$. The authors adapted batch normalization to handle the issue that the different components of the observation with different physical units. As an off-policy algorithm, DDPG learns an actor policy from experiences from an exploration policy by adding noise sampled from a noise process to the actor policy. More than 20 simulated physics tasks of varying difficulty in the MuJoCo environment were solved with the same learning algorithm, network architecture and hyperparameters, and obtained policies with performance competitive with those found by a planning algorithm with full access to the underlying physical model and its derivatives. DDPG can solve problems with 20 times fewer steps of experience than DQN, although it still needs a large number of training episodes to find solutions, as in most model-free RL methods. It is end-to-end, with raw pixels as input. DDPG paper also contains links to videos for illustration.

TRUST REGION POLICY OPTIMIZATION

Schulman et al. (2015) introduced an iterative procedure to monotonically improve policies theoretically, guaranteed by optimizing a surrogate objective function. The authors then proposed a practical algorithm, Trust Region Policy Optimization (TRPO), by making several approximations, including, introducing a trust region constraint, defined by the KL divergence between the new policy and the old policy, so that at every point in the state space, the KL divergence is bounded; approximating the trust region constraint by the average KL divergence constraint; replacing the expectations and Q value in the optimization problem by sample estimates, with two variants: in the single path approach, individual trajectories are sampled; in the vine approach, a rollout set is constructed and multiple actions are performed from each state in the rollout set; and, solving the constrained optimization problem approximately to update the policy's parameter vector. The authors also unified policy iteration and policy gradient with analysis, and showed that policy iteration, policy gradient, and natural policy gradient are special cases of TRPO. In the experiments, TRPO methods performed well on simulated robotic tasks of swimming, hopping, and walking, as well as playing Atari games in an end-to-end manner directly from raw images.

BENCHMARK RESULTS

Duan et al. (2016) presented a benchmark for continuous control tasks, including classic tasks like cart-pole, tasks with very large state and action spaces such as 3D humanoid locomotion and tasks with partial observations, and tasks with hierarchical structure, implemented various algorithms, including batch algorithms: REINFORCE, Truncated Natural Policy Gradient (TNPG), Reward-Weighted Regression (RWR), Relative Entropy Policy Search (REPS), Trust Region Policy Optimization (TRPO), Cross Entropy Method (CEM), Covariance Matrix Adaption Evolution Strategy (CMA-ES); online algorithms: Deep Deterministic Policy Gradient (DDPG); and recurrent variants of batch algorithms. The open source is available at: <https://github.com/rllab/rllab>.

Duan et al. (2016) compared various algorithms, and showed that DDPG, TRPO, and Truncated Natural Policy Gradient (TNPG) (Schulman et al., 2015) are effective in training deep neural network policies, yet better algorithms are called for hierarchical tasks.

3.2.3 COMBINING POLICY GRADIENT AND OFF-POLICY RL

O’Donoghue et al. (2017) proposed to combine policy gradient with off-policy Q-learning (PGQ), to benefit from experience replay. Usually actor-critic methods are on-policy. The authors also showed that action value fitting techniques and actor-critic methods are equivalent, and interpreted regularized policy gradient techniques as advantage function learning algorithms. Empirically, the authors showed that PGQ outperformed DQN and A3C on Atari games.

Nachum et al. (2017) introduced the notion of softmax temporal consistency, to generalize the hard-max Bellman consistency as in off-policy Q-learning, and in contrast to the average consistency as in on-policy SARSA and actor-critic. The authors established the correspondence and a mutual compatibility property between softmax consistent action values and the optimal policy maximizing entropy regularized expected discounted reward. The authors proposed Path Consistency Learning, attempting to bridge the gap between value and policy based RL, by exploiting multi-step path-wise consistency on traces from both on and off policies.

Gu et al. (2017) proposed Q-Prop to take advantage of the stability of policy gradients and the sample efficiency of off-policy RL. Schulman et al. (2017) showed the equivalence between entropy-regularized Q-learning and policy gradient.

3.3 REWARD

Rewards provide evaluative feedbacks for a RL agent to make decisions. Rewards may be sparse so that it is challenging for learning algorithms, e.g., in computer Go, a reward occurs at the end of a game. There are unsupervised ways to harness environmental signals, see Section 4.2. Reward function is a mathematical formulation for rewards. Reward shaping is to modify reward function to facilitate learning while maintaining optimal policy. Reward functions may not be available for some RL problems, which is the focus of this section.

In imitation learning, an agent learns to perform a task from expert demonstrations, with samples of trajectories from the expert, without reinforcement signal, without additional data from the expert while training; two main approaches for imitation learning are behavioral cloning and inverse reinforcement learning. Behavioral cloning, or apprenticeship learning, or learning from demonstration, is formulated as a supervised learning problem to map state-action pairs from expert trajectories to policy, without learning the reward function (Ho et al., 2016; Ho and Ermon, 2016). Inverse reinforcement learning (IRL) is the problem of determining a reward function given observations of optimal behaviour (Ng and Russell, 2000). Abbeel and Ng (2004) approached apprenticeship learning via IRL.

In the following, we discuss learning from demonstration (Hester et al., 2017), and imitation learning with generative adversarial networks (GANs) (Ho and Ermon, 2016; Stadie et al., 2017). We will discuss GANs, a recent unsupervised learning framework, in Section 4.2.3.

Su et al. (2016b) proposed to train dialogue policy jointly with reward model. Christiano et al. (2017) proposed to learn reward function by human preferences from comparisons of trajectory segments. See also Merel et al. (2017); Wang et al. (2017).

LEARNING FROM DEMONSTRATION

Hester et al. (2017) proposed Deep Q-learning from Demonstrations (DQfD) to attempt to accelerate learning by leveraging demonstration data, using a combination of temporal difference (TD), supervised, and regularized losses. In DQfQ, reward signal is not available for demonstration data; however, it is available in Q-learning. The supervised large margin classification loss enables the policy derived from the learned value function to imitate the demonstrator; the TD loss enables the validity of value function according to the Bellman equation and its further use for learning with RL; the regularization loss function on network weights and biases prevents overfitting on small demonstration dataset. In the pre-training phase, DQfD trains only on demonstration data, to obtain

a policy imitating the demonstrator and a value function for continual RL learning. After that, DQfD self-generates samples, and mixes them with demonstration data according to certain proportion to obtain training data. The authors showed that, on Atari games, DQfD in general has better initial performance, more average rewards, and learns faster than DQN.

In AlphaGo (Silver et al., 2016a), to be discussed in Section 5.1.1, the supervised learning policy network is learned from expert moves as learning from demonstration; the results initialize the RL policy network. See also Kim et al. (2014); Pérez-D’Arpino and Shah (2017). See Argall et al. (2009) for a survey of robot learning from demonstration.

GENERATIVE ADVERSARIAL IMITATION LEARNING

With IRL, an agent learns a reward function first, then from which derives an optimal policy. Many IRL algorithms have high time complexity, with a RL problem in the inner loop.

Ho and Ermon (2016) proposed generative adversarial imitation learning algorithm to learn policies directly from data, bypassing the intermediate IRL step. Generative adversarial training was deployed to fit the discriminator, the distribution of states and actions that defines expert behavior, and the generator, the policy.

Generative adversarial imitation learning finds a policy π_θ so that a discriminator \mathcal{D}_R can not distinguish states following the expert policy π_E and states following the imitator policy π_θ , hence forcing \mathcal{D}_R to take 0.5 in all cases and π_θ not distinguishable from π_E in the equilibrium. Such a game is formulated as:

$$\max_{\pi_\theta} \min_{\mathcal{D}_R} -E_{\pi_\theta}[\log \mathcal{D}_R(s)] - E_{\pi_E}[\log(1 - \mathcal{D}_R(s))]$$

The authors represented both π_θ and \mathcal{D}_R as deep neural networks, and found an optimal solution by repeatedly performing gradient updates on each of them. \mathcal{D}_R can be trained with supervised learning with a data set formed from traces from a current π_θ and expert traces. For a fixed \mathcal{D}_R , an optimal π_θ is sought. Hence it is a policy optimization problem, with $-\log \mathcal{D}_R(s)$ as the reward. The authors trained π_θ by trust region policy optimization (Schulman et al., 2015).

THIRD PERSON IMITATION LEARNING

Stadie et al. (2017) argued that previous works in imitation learning, like Ho and Ermon (2016) and Finn et al. (2016b), have the limitation of first person demonstrations, and proposed to learn from unsupervised third person demonstration, mimicking human learning by observing other humans achieving goals.

3.4 MODEL

Model-free RL approaches handle unknown dynamical systems, however, they usually require large number of samples, which may be costly or prohibitive to obtain for real physical systems. Model-based RL approaches learn value function and/or policy in a data-efficient way, however, they may suffer from the issue of model identification so that the estimated models may not be accurate, and the performance is limited by the estimated model.

Chebotar et al. (2017) attempted to combine the advantages of both model-free and model-based RL approaches. The authors focused on time-varying linear-Gaussian policies, and integrated a model-based linear quadratic regulator (LQR) algorithm with a model-free path integral policy improvement algorithm. To generalize the method for arbitrary parameterized policies such as deep neural networks, the authors combined the proposed approach with guided policy search (GPS) (Levine et al., 2016a). The proposed approach does not generate synthetic samples with estimated models to avoid degradation from modelling errors.

See recent work on model-based learning, e.g., Gu et al. (2016b); Henaff et al. (2017); Hester and Stone (2017); Watter et al. (2015).

3.5 PLANNING

Planning constructs a value function or a policy usually with a model, so that planning is usually related to model-based RL methods as discussed in Section 3.4.

Tamar et al. (2016) introduced Value Iteration Networks (VIN), a fully differentiable CNN planning module to approximate the value iteration algorithm, to learn to plan, e.g, policies in RL. In contrast to conventional planning, VIN is model-free, where reward and transition probability are part of the neural network to be learned, so that it avoids issues with system identification. VIN can be trained end-to-end with backpropagation. VIN can generalize in a diverse set of tasks: simple gridworlds, Mars Rover Navigation, continuous control and WebNav Challenge for Wikipedia links navigation (Nogueira and Cho, 2016). One merit of Value Iteration Network, as well as Dueling Network(Wang et al., 2016c), is that they design novel deep neural networks architectures for reinforcement learning problems. See a blog about VIN at <https://github.com/karpathy/paper-notes/blob/master/vin.md>.

Silver et al. (2016b) proposed the predictron to integrate learning and planning into one end-to-end training procedure with raw input in Markov reward process, which can be regarded as Markov decision process without actions. See classical Dyna-Q (Sutton, 1990).

3.6 EXPLORATION

a RL agent usually uses exploration to reduce its uncertainty about the reward function and transition probabilities of the environment. In tabular cases, this uncertainty can be quantified as confidence intervals or posterior of environment parameters, which are related to the state-action visit counts. With count-based exploration, a RL agent uses visit counts to guide its behaviour to reduce uncertainty. However, count-based methods are not directly useful in large domains. Intrinsic motivation suggests to explore what is surprising, typically in learning process based on change in prediction error. Intrinsic motivation methods do not require Markov property and tabular representation as count-based methods require. Bellemare et al. (2016) proposed pseudo-count, a density model over the state space, to unify count-based exploration and intrinsic motivation, by introducing information gain, to relate to confidence intervals in count-based exploration, and to relate to learning progress in intrinsic motivation. The author established pseudo-count's theoretical advantage over previous intrinsic motivation methods, and validated it with Atari games.

Nachum et al. (2017) proposed an under-appreciated reward exploration technique to avoid the previous ineffective, undirected exploration strategies of the reward landscape, as in ϵ -greedy and entropy regularization, and to promote directed exploration of the regions, in which the log-probability of an action sequence under the current policy under-estimates the resulting reward. The under-appreciated reward exploration strategy resulted from importance sampling from the optimal policy, and combined a mode seeking and a mean seeking terms to tradeoff exploration and exploitation. The authors implemented the proposed exploration strategy with minor modifications to REINFORCE, and validated it, for the first time with a RL method, on several algorithmic tasks.

Osband et al. (2016) proposed bootstrapped DQN to combine deep exploration with deep neural networks to achieve efficient learning. Houthooft et al. (2016) proposed variational information maximizing exploration for continuous state and action spaces. Fortunato et al. (2017) proposed NoisyNet for efficient exploration by adding parametric noise added to weights of deep neural networks.

4 IMPORTANT MECHANISMS

In this section, we discuss important mechanisms for the development of (deep) reinforcement learning, including attention and memory, unsupervised learning, transfer learning, semi-supervised learning, hierarchical RL, and learning to learn.

4.1 ATTENTION AND MEMORY

Attention is a mechanism to focus on the salient parts. Memory provides data storage for long time, and attention is an approach for memory addressing.

Mnih et al. (2014) applied attention to image classification and object detection. Xu et al. (2015) integrated attention to image captioning. We briefly discuss application of attention in computer vision in Section 5.4. The attention mechanism is also deployed in NLP, e.g., in Bahdanau et al. (2015; 2017), and with external memory, in differentiable neural computer (Graves et al., 2016). Most works follow a soft attention mechanism (Bahdanau et al., 2015), a weighted addressing scheme to all memory locations. There are endeavours for hard attention (Zaremba and Sutskever, 2015; Xu et al., 2015; Luo et al., 2016; Gulcehre et al., 2016), which is the way conventional computers access memory.

See recent work on attention and/or memory, e.g., Ba et al. (2014; 2016); Chen et al. (2016a); Danihelka et al. (2016); Duan et al. (2017); Eslami et al. (2016); Gregor et al. (2015); Jaderberg et al. (2015); Kaiser and Bengio (2016); Kadlec et al. (2016); Luo et al. (2016); Oh et al. (2016); Oquab et al. (2015); Vaswani et al. (2017); Weston et al. (2015); Sukhbaatar et al. (2015); Yang et al. (2015); Zagoruyko and Komodakis (2017); Zaremba and Sutskever (2015). See <http://distill.pub/2016/augmented-rnns/> and <http://www.wildml.com/2016/01/attention-and-memory-in-deep-learning-and-nlp/> for blogs about attention and memory.

4.1.1 DIFFERENTIABLE NEURAL COMPUTER

Graves et al. (2016) proposed differentiable neural computer (DNC), in which, a neural network can read from and write to an external memory, so that DNC can solve complex, structured problems, which a neural network without read-write memory can not solve. DNC minimizes memory allocation interference and enables long-term storage. Similar to a conventional computer, in a DNC, the neural network is the controller and the external memory is the random-access memory; and a DNC represents and manipulates complex data structures with the memory. Differently, a DNC learns such representation and manipulation end-to-end with gradient descent from data in a goal-directed manner. When trained with supervised learning, a DNC can solve synthetic question answering problems, for reasoning and inference in natural language; it can solve the shortest path finding problem between two stops in transportation networks and the relationship inference problem in a family tree. When trained with reinforcement learning, a DNC can solve a moving blocks puzzle with changing goals specified by symbol sequences. DNC outperformed normal neural network like LSTM or DNC’s precursor Neural Turing Machine (Graves et al., 2014); with harder problems, an LSTM may simply fail. Although these experiments are relatively small-scale, we expect to see further improvements and applications of DNC. See Deepmind’s description of DNC at <https://deepmind.com/blog/differentiable-neural-computers/>.

4.2 UNSUPERVISED LEARNING

Unsupervised learning is a way to take advantage of the massive amount of data, and would be a critical mechanism to achieve general artificial intelligence. Unsupervised learning is categorized into non-probabilistic models, like sparse coding, autoencoders, k-means etc, and probabilistic (generative) models, where density functions are concerned, either explicitly or implicitly (Salakhutdinov, 2016). Among probabilistic (generative) models with explicit density functions, some are with tractable models, like fully observable belief nets, neural autoregressive distribution estimators, and PixelRNN, etc; some are with non-tractable models, like Boltzmann machines, variational autoencoders, Helmholtz machines, etc. For probabilistic (generative) models with implicit density functions, we have generative adversarial networks, moment matching networks, etc.

In the following, we discuss Horde (Sutton et al., 2011), and unsupervised auxiliary learning (Jaderberg et al., 2017), two ways to take advantages of possible non-reward training signals in environments. We also discuss generative adversarial networks (Goodfellow et al., 2014). See also Le et al. (2012), Chen et al. (2016), Liu et al. (2017).

4.2.1 HORDE

Sutton et al. (2011) proposed to represent knowledge with general value function, where policy, termination function, reward function, and terminal reward function are parameters. The authors then proposed Horde, a scalable real-time architecture for learning in parallel general value functions for independent sub-agents from unsupervised sensorimotor interaction, i.e., nonreward signals and observations. Horde can learn to predict the values of many sensors, and policies to maximize those

sensor values, with general value functions, and answer predictive or goal-oriented questions. Horde is off-policy, i.e., it learns in real-time while following some other behaviour policy, and learns with gradient-based temporal difference learning methods, with constant time and memory complexity per time step.

4.2.2 UNSUPERVISED AUXILIARY LEARNING

Environments may contain abundant possible training signals, which may help to expedite achieving the main goal of maximizing the accumulative rewards, e.g., pixel changes may imply important events, and auxiliary reward tasks may help to achieve a good representation of rewarding states. This may be even helpful when the extrinsic rewards are rarely observed.

Jaderberg et al. (2017) proposed UNsupervised REinforcement and Auxiliary Learning (UNREAL) to improve learning efficiency by maximizing pseudo-reward functions, besides the usual cumulative reward, while sharing a common representation. UNREAL is composed of RNN-LSTM base agent, pixel control, reward prediction, and value function replay. The base agent is trained on-policy with A3C (Mnih et al., 2016). Experiences of observations, rewards and actions are stored in a replay buffer, for being used by auxiliary tasks. The auxiliary policies use the base CNN and LSTM, together with a deconvolutional network, to maximize changes in pixel intensity of different regions of the input images. The reward prediction module predicts short-term extrinsic reward in next frame by observing the last three frames, to tackle the issue of reward sparsity. Value function replay further trains the value function. UNREAL improved A3C’s performance on Atari games, and performed well on 3D Labyrinth game. UNREAL has a shared representation among signals, while Horde trains each value function separately with distinct weights. See Deepmind’s description of UNREAL at <https://deepmind.com/blog/reinforcement-learning-unsupervised-auxiliary-tasks/>.

We discuss robotics navigation with similar unsupervised auxiliary learning (Mirowski et al., 2017) in Section 5.2. See also Lample and Chaplot (2016).

4.2.3 GENERATIVE ADVERSARIAL NETWORKS

Goodfellow et al. (2014) proposed generative adversarial nets (GANs) to estimate generative models via an adversarial process by training two models simultaneously, a generative model G to capture the data distribution, and a discriminative model D to estimate the probability that a sample comes from the training data but not the generative model G .

Goodfellow et al. (2014) modelled G and D with multilayer perceptrons: $G(z : \theta_g)$ and $D(x : \theta_d)$, where θ_g and θ_d are parameters, x are data points, and z are input noise variables. Define a prior on input noise variable $p_z(z)$. G is a differentiable function and $D(x)$ outputs a scalar as the probability that x comes from the training data rather than p_g , the generative distribution we want to learn.

D will be trained to maximize the probability of assigning labels correctly to samples from both training data and G . Simultaneously, G will be trained to minimize such classification accuracy, $\log(1 - D(G(z)))$. As a result, D and G form the two-player minimax game as follows:

$$\min_G \max_D E_{x \sim p_{data}(x)} [\log D(x)] + E_{z \sim p_z(z)} [\log(1 - D(G(z)))]$$

Goodfellow et al. (2014) showed that as G and D are given enough capacity, generative adversarial nets can recover the data generating distribution, and provided a training algorithm with backpropagation by minibatch stochastic gradient descent.

See Goodfellow (2017) for Ian Goodfellow’s summary of his NIPS 2016 Tutorial on GANs. GANs have received much attention and many works have been appearing after the tutorial.

GANs are notoriously hard to train. See Arjovsky et al. (2017) for Wasserstein GAN (WGAN) as a stable GANs model. Gulrajani et al. (2017) proposed to improve stability of WGAN by penalizing the norm of the gradient of the discriminator with respect to its input, instead of clipping weights as in Arjovsky et al. (2017). Mao et al. (2016) proposed Least Squares GANs (LSGANs), another stable model. Berthelot et al. (2017) proposed BEGAN to improve WGAN by an equilibrium enforcing model, and set a new milestone in visual quality for image generation. Bellemare et al. (2017) proposed Cramér GAN to satisfy three machine learning properties of probability divergences: sum

invariance, scale sensitivity, and unbiased sample gradients. Hu et al. (2017) unified GANs and Variational Autoencoders (VAEs).

We discuss imitation learning with GANs in Section 3.3, including generative adversarial imitation learning, and third person imitation learning. Finn et al. (2016a) established a connection between GANs, inverse RL, and energy-based models. Pfau and Vinyals (2016) established the connection between GANs and actor-critic algorithms.

4.3 TRANSFER LEARNING

Transfer learning is about transferring knowledge learned from different domains, possibly with different feature spaces and/or different data distributions (Taylor and Stone, 2009; Pan and Yang, 2010; Weiss et al., 2016). As reviewed in Pan and Yang (2010), transfer learning can be inductive, transductive, or unsupervised; inductive transfer learning includes self-taught learning and multi-task learning; and transductive transfer learning includes domain adaptation and sample selection bias/covariance shift.

Gupta et al. (2017) formulated the multi-skill problem for two agents to learn multiple skills, defined the common representation using which to map states and to project the execution of skills, and designed an algorithm for two agents to transfer the informative feature space maximally to transfer new skills, with similarity loss metric, autoencoder, and reinforcement learning. The authors validated their proposed approach with two simulated robotic manipulation tasks.

See also recent work in transfer learning e.g., Andreas et al. (2017); Dong et al. (2015); Ganin et al. (2016); Kaiser et al. (2017a); Kansky et al. (2017); Long et al. (2015; 2016); Maurer et al. (2016); Mo et al. (2016); Parisotto et al. (2016); Papernot et al. (2017); Pérez-D’Arpino and Shah (2017); Rajendran et al. (2017); Whye Teh et al. (2017); Yosinski et al. (2014). See NIPS 2015 Transfer and Multi-Task Learning: Trends and New Perspectives Workshop.

4.4 SEMI-SUPERVISED LEARNING

In semi-supervised learning, both labelled and unlabelled data are used to train a model to improve performance (Zhu and Goldberg, 2009).

Finn et al. (2017) proposed semi-supervised reinforcement learning, to learn both with labeled Markov decision processes (MDPs) with reward functions, and with unlabeled MDPs without reward functions. Audiffren et al. (2015) proposed semi-supervised inverse reinforcement learning, to learn both with expert’s trajectories, and unsupervised trajectories which may not be performed by experts. Finn et al. (2017) and Audiffren et al. (2015) applied semi-supervised learning to RL with different ways of defining the labels. See also Cheng et al. (2016); Dai et al. (2017); Kingma et al. (2014); Papernot et al. (2017); Yang et al. (2017).

4.5 HIERARCHICAL REINFORCEMENT LEARNING

Hierarchical RL is a way to learn, plan, and represent knowledge with spatio-temporal abstraction at multiple levels. Hierarchical RL is an approach for issues of sparse rewards and/or long horizons.

Vezhnevets et al. (2016) proposed strategic attentive writer (STRAW), a deep recurrent neural network architecture, for learning high-level temporally abstracted macro-actions in an end-to-end manner based on observations from the environment. Macro-actions are sequences of actions commonly occurring. STRAW builds a multi-step action plan, updated periodically based on observing rewards, and learns for how long to commit to the plan by following it without replanning. STRAW learns to discover macro-actions automatically from data, in contrast to the manual approach in previous work. Vezhnevets et al. (2016) validated STRAW on next character prediction in text, 2D maze navigation, and Atari games.

Kulkarni et al. (2016) proposed hierarchical-DQN (h-DQN) by organizing goal-driven intrinsically motivated deep RL modules hierarchically to work at different time-scales. h-DQN integrates a top level action value function and a lower level action value function; the former learns a policy over intrinsic sub-goals, or options (Sutton et al., 1999); the latter learns a policy over raw actions

to satisfy given sub-goals. In a hard Atari game, Montezuma’s Revenge, h-DQN outperformed previous methods, including DQN and A3C.

Florensa et al. (2017) proposed to pre-train a large span of skills using Stochastic Neural Networks with an information-theoretic regularizer, then on top of these skills, to train high-level policies for downstream tasks. Pre-training is based on a proxy reward signal, which is a form of intrinsic motivation to explore agent’s own capabilities; its design requires minimal domain knowledge about the downstream tasks. Their method combined hierarchical methods with intrinsic motivation, and the pre-training follows an unsupervised way.

Tessler et al. (2017) proposed a hierarchical deep RL network architecture for lifelong learning. Reusable skills, or sub-goals, are learned to transfer knowledge to new tasks. The authors tested their approach on the game of Minecraft.

See also Machado et al. (2017), Peng et al. (2017a), Schaul et al. (2015), Sharma et al. (2017), Vezhnevets et al. (2017), Yao et al. (2014). See a survey on hierarchical RL (Barto and Mahadevan, 2003).

4.6 LEARNING TO LEARN

Learning to learn is about learning to adapt rapidly to new tasks. It is related to transfer learning, multi-task learning, representation learning, meta learning, and one/few/zero-shot learning. It is a core ingredient to achieve strong AI (Lake et al., 2016).

Duan et al. (2017) and Wang et al. (2016a) proposed to learn a flexible RNN model to handle a family of RL tasks, to improve sample efficiency, learn new tasks in a few samples, and benefit from prior knowledge. The agent is modelled with RNN, with inputs of observations, rewards, actions and termination flags; the weights of RNN are trained with RL, TRPO in Duan et al. (2017) and A3C in Wang et al. (2016a), respectively, and achieve similar performance for various problems to specific RL algorithms. Duan et al. (2017) experimented with multi-arm bandits, tabular MDPs and visual navigation, and discussed that, for larger problems, better RL algorithms are needed to train RNN. Wang et al. (2016a) experimented with bandits with independent arms, bandits with dependant arms, restless arms and MDPs. A future work is to improve scalability.

Lake et al. (2015) proposed an one-shot concept learning model, for handwritten characters in particular, with probabilistic program induction. Koch et al. (2015) proposed siamese neural networks with metric learning for one-shot image recognition. Vinyals et al. (2016) designed matching networks for one-shot classification. Duan et al. (2017) proposed a model for one-shot imitation learning with attention for robotics. Ravi and Larochelle (2017) proposed a meta-learning model for few shot learning. Li and Malik (2017) proposed to automate unconstrained continuous optimization algorithms with guided policy search (Levine et al., 2016a) by representing a particular optimization algorithm as a policy, and convergence rate as reward. Johnson et al. (2016) presented zero-shot translation for Google’s multilingual neural machine translation system. Kaiser et al. (2017b) designed a large scale memory module for life-long one-shot learning to remember rare events. Kansky et al. (2017) proposed Schema Networks for zero-shot transfer with a generative causal model of intuitive physics. Snell et al. (2017) proposed prototypical networks for few/zero-shot classification by learning a metric space to compute distances to prototype representations of each class. See Ruder (2017) for an overview about multi-task learning.

5 APPLICATIONS

Reinforcement learning has a wide range of applications. We discuss games in Section 5.1 and robotics in Section 5.2, two classical RL application areas. Games will still be an important testbed for AI. Robotics will be critical in the era of AI. Next we discuss natural language processing in Section 5.3, which enjoys wide and deep applications of RL recently. Computer vision follows in Section 5.4, in which, there are efforts for integration of vision and language. Neural architecture design in Section 5.5 is an exciting, new application of RL. In Section 5.6, we discuss business management, like ads, recommendation, customer management, and marketing. We discuss finance in Section 5.7. Business and finance have natural problems for RL. We discuss healthcare in Section 5.8, which receives much attention recently, esp. after the success of deep learning. We discuss

Industry 4.0 in Section 5.9. Many countries have made plans to integrate AI with manufacturing. We discuss smart grid in Section 5.10, intelligent transportation systems in Section 5.11, and computer systems in Section 5.12. There are optimization and control problems in these areas, and many of them are concerned with networking and graphs. These application areas may overlap with each other, e.g., a robot may need skills for many or even all of the application areas.

Reinforcement learning is widely used in operations research (Powell, 2011), e.g., supply chain, inventory management, resource management, etc; we do not list it as an application area — it is implicitly a component in application areas like intelligent transportation system and Industry 4.0. We do not list smart city, an important application area of AI, as it includes several application areas here: healthcare, intelligent transportation system, smart grid, etc. We do not discuss some interesting applications, like music generation (Jaques et al., 2017), and retrosynthesis (Segler et al., 2017). See previous work on lists of RL applications at: <http://bit.ly/2pDEs1Q>, and <http://bit.ly/2rjsmaz>. We may only touch the surface of some application areas. It is desirable to do a deeper analysis of all application areas listed in the following, which we leave as a future work.

5.1 GAMES

Games provide excellent testbeds for RL/AI algorithms. We discuss Deep Q-Network (DQN) in Section 3.1.1 and its extensions, all of which experimented with Atari games. We discuss Mnih et al. (2016) in Section 3.2.1, Jaderberg et al. (2017) in Section 4.2, and Mirowski et al. (2017) in Section 5.2, and they used Labyrinth as the testbed. See Yannakakis and Togelius (2017) for a draft book on artificial intelligence and games.

Backgammon and computer Go are perfect information board games. In Section 5.1.1, we discuss briefly Backgammon, and focus on computer Go, in particular, AlphaGo. Variants of card games, including majiang/mahjong, are imperfect information board games, which we discuss in Section 5.1.2, and focus on Texas Hold'em Poker. In video games, information may be perfect or imperfect, and game theory may be deployed or not. We discuss video games in Section 5.1.3. We will see more achievements in imperfect information games and video games, and their applications.

5.1.1 PERFECT INFORMATION BOARD GAMES

Board games like Backgammon, Go, chess, checker and Othello, are classical testbeds for RL/AI algorithms. In such games, players reveal perfect information. Tesauro (1994) approached Backgammon by using neural networks to approximate value function learned with TD learning, and achieved human level performance. We focus on computer Go, in particular, AlphaGo (Silver et al., 2016a), for its significance.

COMPUTER GO

The challenge of solving Computer Go comes from not only the gigantic search space of size 250^{150} , an astronomical number, but also the hardness of position evaluation (Müller, 2002), which was successfully used in solving many other games, like Backgammon and chess.

AlphaGo (Silver et al., 2016a), a computer Go program, won the human European Go champion, 5 games to 0, in October 2015, and became the first computer Go program to win a human professional Go player without handicaps on a full-sized 19×19 board. Soon after that in March 2016, AlphaGo defeated Lee Sedol, an 18-time world champion Go player, 4 games to 1, making headline news worldwide. This set a landmark in AI. AlphaGo defeated Ke Jie 3:0 in May 2017.

ALPHAGO: TRAINING PIPELINE AND MCTS

We discuss briefly how AlphaGo works based on Silver et al. (2016a) and Sutton and Barto (2017). See Sutton and Barto (2017) for a detailed and intuitive description of AlphaGo. See Deepmind's description of AlphaGo at goo.gl/IzoQ1d.

AlphaGo was built with techniques of deep CNN, supervised learning, reinforcement learning, and Monte Carlo tree search (MCTS) (Browne et al., 2012; Gelly et al., 2012). AlphaGo is composed of two phases: neural network training pipeline and MCTS. The training pipeline phase includes

training a supervised learning (SL) policy network from expert moves, a fast rollout policy, a RL policy network, and a RL value network.

The SL policy network has convolutional layers, ReLU nonlinearities, and an output softmax layer representing probability distribution over legal moves. The inputs to the CNN are $19 \times 19 \times 48$ image stacks, where 19 is the dimension of a Go board and 48 is the number of features. State-action pairs are sampled from expert moves to train the network with stochastic gradient ascent to maximize the likelihood of the move selected in a given state. The fast rollout policy uses a linear softmax with small pattern features.

The RL policy network improves SL policy network, with the same network architecture, and the weights of SL policy network as initial weights, and policy gradient for training. The reward function is +1 for winning and -1 for losing in the terminal states, and 0 otherwise. Games are played between the current policy network and a random, previous iteration of the policy network, to stabilize the learning and to avoid overfitting. Weights are updated by stochastic gradient ascent to maximize the expected outcome.

The RL value network still has the same network architecture as SL policy network, except the output is a single scalar predicting the value of a position. The value network is learned in a Monte Carlo policy evaluation approach. To tackle the overfitting problem caused by strongly correlated successive positions in games, data are generated by self-play between the RL policy network and itself until game termination. The weights are trained by regression on state-outcome pairs, using stochastic gradient descent to minimize the mean squared error between the prediction and the corresponding outcome.

In MCTS phase, AlphaGo selects moves by lookahead search. It builds a partial game tree starting from the current state, in the following stages: 1) select a promising node to explore further, 2) expand a leaf node guided by the SL policy network and collected statistics, 3) evaluate a leaf node with a mixture of the RL value network and the rollout policy, 4) backup evaluations to update the action values. A move is then selected.

DISCUSSIONS

The Deepmind team integrated several existing techniques together to engineer AlphaGo and it has achieved tremendous results. However, the RL policy network and RL value network are not strong/accurate enough, so that the RL value network, together with the SL policy network and the rollout network, assist MCTS to search for the move. This might explain the one game loss against Lee Sedol. The 2017 version of AlphaGo vs. Ke Jie worked on a single machine with TPU, and our guess is that it improved the accuracy of policy network and value network by self play so that it needs less search with MCTS. Moreover, AlphaGo still requires manually defined features with human knowledge, so it is not entirely an end-to-end solution yet; in contrast, DQN requires only raw pixels and scores as inputs. Such a room for improvements would inspire intellectual inquisition for better computer Go programs, potentially with deep RL only, without MCTS, like TD-Gammon (Sutton and Barto, 2017). This would be based on a novel RL algorithm, a novel deep neural network architecture, and powerful computation. New RL algorithms are called for, possibly for better reasoning. New deep neural network architectures are called for, for the sophistication to represent complex scenarios in Go and the elegance for learning in a reasonable time, so that an optimal policy and/or an optimal value function can be directly approximated to make decisions without the help of MCTS to choose moves. Admittedly, such endeavour would be illusive at large currently. See Wang et al. (2017) for an endeavour in this direction.

Being more practical, we expect more applications/extensions of techniques in Silver et al. (2016a) in solving problems requiring titanic search spaces, like classical AI problems, e.g., planning, scheduling, and constraint satisfaction, etc.¹

¹Andrej Karpathy posted a blog titled "AlphaGo, in context", after AlphaGo defeated Ke Jie in May 2017. He characterized properties of Computer Go as: fully deterministic, fully observable, discrete action space, accessible perfect simulator, relatively short episode/game, clear and fast evaluation conducive for many trial-and-errors, and huge datasets of human play games, to illustrate the narrowness of AlphaGo. It is true that computer Go has limitations in the problem setting and thus potential applications, and is far from artificial general intelligence. However, we see the success of AlphaGo as the triumph of AI, in particular, AlphaGo's underlying techniques, i.e., learning from demonstration (as supervised learning), deep learning, reinforcement

5.1.2 IMPERFECT INFORMATION BOARD GAMES

Imperfect information games, or game theory in general, have many applications, e.g., security and medical decision support (Sandholm, 2015). It is interesting to see more progress of deep RL in such applications, and the full version of Texas Hold'em.

Heinrich and Silver (2016) proposed Neural Fictitious Self-Play (NFSP) to combine fictitious self-play with deep RL to learn approximate Nash equilibria for games of imperfect information in a scalable end-to-end approach without prior domain knowledge. NFSP was evaluated on two-player zero-sum games. In Leduc poker, NFSP approached a Nash equilibrium, while common RL methods diverged. In Limit Texas Hold'em, a real-world scale imperfect-information game, NFSP performed similarly to state-of-the-art, superhuman algorithms which are based on significant domain expertise.

Heads-up Limit Hold'em Poker was essentially solved (Bowling et al., 2015) with counterfactual regret minimization (CFR), which is an iterative method to approximate a Nash equilibrium of an extensive-form game with repeated self-play between two regret-minimizing algorithms.

DEEPSTACK

Recently, significant progress has been made for Heads-up No-Limit Hold'em Poker (Moravčík et al., 2017), the DeepStack computer program defeated professional poker players for the first time. DeepStack utilized the recursive reasoning of CFR to handle information asymmetry, focusing computation on specific situations arising when making decisions and use of value functions trained automatically, with little domain knowledge or human expert games, without abstraction and offline computation of complete strategies as before as in Sandholm (2015).

5.1.3 VIDEO GAMES

Video games would be great testbeds for artificial general intelligence.

Wu and Tian (2017) deployed A3C with CNN to train an agent in a partially observable 3D environment, Doom, from recent four raw frames and game variables, to predict next action and value function, following the curriculum learning (Bengio et al., 2009) approach of starting with simple tasks and gradually transition to harder ones. It is nontrivial to apply A3C to such 3D games directly, partly due to sparse and long term reward. The authors won the champion in Track 1 of ViZDoom Competition by a large margin, and plan the following future work: a map from an unknown environment, localization, a global plan to act, and visualization of the reasoning process.

Dosovitskiy and Koltun (2017) approached the problem of sensorimotor control in immersive environments with supervised learning, and won the Full Deathmatch track of the Visual Doom AI Competition. We list it here since it is usually a RL problem, yet it was solved with supervised learning. Lample and Chaplot (2016) also discussed how to tackle Doom.

Peng et al. (2017b) proposed a multiagent actor-critic framework, with a bidirectionally-coordinated network to form coordination among multiple agents in a team, deploying the concept of dynamic grouping and parameter sharing for better scalability. The authors used StarCraft as the testbed. Without human demonstration or labelled data as supervision, the proposed approach learned strategies for coordination similar to the level of experienced human players, like move without collision, hit and run, cover attack, and focus fire without overkill. Usunier et al. (2017); Justesen and Risi (2017) also studied StarCraft.

Oh et al. (2016) and Tessler et al. (2017) studied Minecraft, Chen and Yi (2017); Firoiu et al. (2017) studied Super Smash Bros, and Kansky et al. (2017) proposed Schema Networks and empirically studied variants of Breakout in Atari games.

See Ontañón et al. (2013) for a survey about Starcraft. Check AIIDE Starcraft AI Competitions, and its history at <https://www.cs.mun.ca/~dchurchill/starcraftaicomp/history.shtml>

learning, and Monte Carlo tree search; these techniques are present in many recent achievements in AI. As a whole technique, AlphaGo will probably shed lights on classical AI areas, like planning, scheduling, and constraint satisfaction (Silver et al., 2016a), and new areas for AI, like retrosynthesis (Segler et al., 2017). Reportedly, the success of AlphaGo's conquering titanic search space inspired quantum physicists to solve the quantum many-body problem (Carleo and Troyer, 2017).

5.2 ROBOTICS

Robotics is a classical area for reinforcement learning. See Kober et al. (2013) for a survey of RL in robotics, Deisenroth et al. (2013) for a survey on policy search for robotics, and Argall et al. (2009) for a survey of robot learning from demonstration. See the journal *Science Robotics*. It is interesting to note that from NIPS 2016 invited talk, Boston Dynamics robots did not use machine learning.

In the following, we discuss guided policy search (Levine et al., 2016a) and learn to navigate (Mirowski et al., 2017). See more recent robotics papers, e.g., Chebotar et al. (2016; 2017); Duan et al. (2017); Finn and Levine (2016); Gu et al. (2016a); Lee et al. (2017); Levine et al. (2016b); Mahler et al. (2017); Pérez-D’Arpino and Shah (2017); Popov et al. (2017); Yahya et al. (2016); Zhu et al. (2016).

5.2.1 GUIDED POLICY SEARCH

Levine et al. (2016a) proposed to train the perception and control systems jointly end-to-end, to map raw image observations directly to torques at the robot’s motors. The authors introduced guided policy search (GPS) to train policies represented as CNN, by transforming policy search into supervised learning to achieve data efficiency, with training data provided by a trajectory-centric RL method operating under unknown dynamics. GPS alternates between trajectory-centric RL and supervised learning, to obtain the training data coming from the policy’s own state distribution, to address the issue that supervised learning usually does not achieve good, long-horizon performance. GPS utilizes pre-training to reduce the amount of experience data to train visuomotor policies. Good performance was achieved on a range of real-world manipulation tasks requiring localization, visual tracking, and handling complex contact dynamics, and simulated comparisons with previous policy search methods. As the authors mentioned, “this is the first method that can train deep visuomotor policies for complex, high-dimensional manipulation skills with direct torque control”.

5.2.2 LEARN TO NAVIGATE

Mirowski et al. (2017) obtained the navigation ability by solving a RL problem maximizing cumulative reward and jointly considering un/self-supervised tasks to improve data efficiency and task performance. The authors addressed the sparse reward issues by augmenting the loss with two auxiliary tasks, 1) unsupervised reconstruction of a low-dimensional depth map for representation learning to aid obstacle avoidance and short-term trajectory planning; 2) self-supervised loop closure classification task within a local trajectory. The authors incorporated a stacked LSTM to use memory at different time scales for dynamic elements in the environments. The proposed agent learn to navigate in complex 3D mazes end-to-end from raw sensory input, and performed similarly to human level, even when start/goal locations change frequently.

In this approach, navigation is a by-product of the goal-directed RL optimization problem, in contrast to conventional approaches such as Simultaneous Localisation and Mapping (SLAM), where explicit position inference and mapping are used for navigation. This may have the chance to replace the popular SLAM, which usually requires manual processing.

5.3 NATURAL LANGUAGE PROCESSING

In the following we talk about natural language processing (NLP), dialogue systems in Section 5.3.1, machine translation in Section 5.3.2, and text generation in Section 5.3.3. There are many interesting issues in NLP, and we list some in the following.

- language tree-structure learning, e.g., Socher et al. (2011; 2013); Yogatama et al. (2017)
- question answering, e.g., Celikyilmaz et al. (2017)?, Shen et al. (2017), Trischler et al. (2016), Xiong et al. (2017a), and Wang et al. (2017a), Choi et al. (2017)
- summarization, e.g., Paulus et al. (2017)
- sentiment analysis (Liu, 2012), e.g., Radford et al. (2017)
- information retrieval (Manning et al., 2008), e.g., Zhang et al. (2016), and Mitra and Craswell (2017)
- information extraction, e.g., Narasimhan et al. (2016)

- automatic query reformulation, e.g., Nogueira and Cho (2017)
- language to executable program, e.g., Guu et al. (2017)
- text games, e.g., Wang et al. (2016b), He et al. (2016b), and Narasimhan et al. (2015)

Deep learning has been permeating into many subareas in NLP, and helping make significant progress. The above is a partial list. It appears that NLP is still a field, more about synergy than competition, for deep learning vs. non-deep learning algorithms, and for approaches based on no domain knowledge (end-to-end) vs linguistics knowledge. Some non-deep learning algorithms are effective and perform well, e.g., word2vec (Mikolov et al., 2013) and fastText (Joulin et al., 2017), and many works that study syntax and semantics of languages, see a recent example in semantic role labeling (He et al., 2017b). Some deep learning approaches to NLP problems incorporate explicitly or implicitly linguistics knowledge, e.g., Socher et al. (2011; 2013); Yogatama et al. (2017). See an article by Christopher D. Manning, titled "Last Words: Computational Linguistics and Deep Learning, A look at the importance of Natural Language Processing", at <http://mitp.nautil.us/article/170/last-words-computational-linguistics-and-deep-learning>. See conferences like ACL, EMNLP, and NAACL for papers about NLP.

5.3.1 DIALOGUE SYSTEMS

In dialogue systems, conversational agents, or simply, chatbots, human and computer interacts with natural language. We intentionally remove "spoken" before "dialogue systems" to accommodate both spoken and written language user interface (UI). There are usually four categories: social chatbots, infobots (interactive question answering), task completion bots (task-oriented or goal-oriented) and personal assistant bots (Deng, 2017). We have seen generation one dialogue systems: symbolic rule/template based, and generation two: data driven with (shallow) learning. We are now experiencing generation three: data driven with deep learning, and reinforcement learning usually play an important role. A dialogue system usually include the following modules: (spoken) language understanding, dialogue manager (dialogue state tracker and dialogue policy learning), and a natural language generation (Young et al., 2013). In task-oriented systems, there is usually a knowledge base to query. A deep learning approach, as usual, attempts to make the learning of the system parameters end-to-end. See Deng (2017) for more details. See a survey paper on applying machine learning to speech recognition (Deng and Li, 2013).

Li et al. (2017b) presented an end-to-end task-completion neural dialogue system with parameters learned by supervised and reinforcement learning. The proposed framework includes a user simulator (Li et al., 2016d) and a neural dialogue system. The user simulator consists of user agenda modelling and natural language generation. The neural dialogue system is composed of language understanding and dialogue management (dialogue state tracking and policy learning). The authors deployed RL to train dialogue management end-to-end, representing the dialogue policy as a deep Q-network (Mnih et al., 2015), with the tricks of a target network and a customized experience replay, and using a rule-based agent to warm-start the system with supervised learning. The source code is available at <http://github.com/MiuLab/TC-Bot>.

Dhingra et al. (2017) proposed KB-InfoBot, a goal-oriented dialogue system for multi-turn information access. KB-InfoBot is trained end-to-end using RL from user feedback with differentiable operations, including those for accessing external knowledge database (KB). In previous work, e.g., Li et al. (2017b) and Wen et al. (2017), a dialogue system accesses real world knowledge from KB by symbolic, SQL-like operations, which is non-differentiable and disables the dialogue system from fully end-to-end trainable. KB-InfoBot achieved the differentiability by inducing a soft posterior distribution over the KB entries to indicate which ones the user is interested in. The authors designed a modified version of the episodic REINFORCE algorithm to explore and learn both the policy to select dialogue acts and the posterior over the KB entries for correct retrievals. The authors deployed imitation learning from rule-based belief trackers and policy to warm up the system.

Su et al. (2016b) proposed an on-line learning framework to train the dialogue policy jointly with the reward model via active learning with a Gaussian process model, to tackle the issue that it is unreliable and costly to use explicit user feedback as the reward signal. The authors showed empirically that the proposed framework reduced manual data annotations significantly and mitigated noisy user feedback in dialogue policy learning.

Li et al. (2016c) proposed to use deep RL to generate dialogues to model future reward for better informativity, coherence, and ease of answering, to attempt to address the issues in the sequence to sequence models based on Sutskever et al. (2014): the myopia and misalignment of maximizing the probability of generating a response given the previous dialogue turn, and the infinite loop of repetitive responses. The authors designed a reward function to reflect the above desirable properties, and deployed policy gradient to optimize the long term reward. It would be interesting to investigate the reward model with the approach in Su et al. (2016b) or with inverse RL and imitation learning as discussed in Section 3.3, although Su et al. (2016b) mentioned that such methods are costly, and humans may not act optimally.

Some recent papers follow: Asri et al. (2016), Bordes et al. (2017), Chen et al. (2016b), Eric and Manning (2017), Fatemi et al. (2016), Kandasamy et al. (2017), Lewis et al. (2017), Li et al. (2016a), Li et al. (2017a), Li et al. (2017b), Lipton et al. (2016), Mesnil et al. (2015), Mo et al. (2016), Peng et al. (2017a), Saon et al. (2016), Shah et al. (2016), She and Chai (2017), Su et al. (2016a), Weiss et al. (2017), Wen et al. (2015a), Wen et al. (2017), Williams and Zweig (2016), Williams et al. (2017), Xiong et al. (2017b), Yang et al. (2016), Zhang et al. (2017a), Zhang et al. (2017c), Zhao and Eskenazi (2016), Zhou et al. (2017). See Serban et al. (2015) for a survey of corpora for building dialogue systems.

See NIPS 2016 Workshop on End-to-end Learning for Speech and Audio Processing, and NIPS 2015 Workshop on Machine Learning for Spoken Language Understanding and Interactions.

5.3.2 MACHINE TRANSLATION

Neural machine translation (Kalchbrenner and Blunsom, 2013; Cho et al., 2014; Sutskever et al., 2014; Bahdanau et al., 2015) utilizes end-to-end deep learning for machine translation, and becomes dominant, against the traditional statistical machine translation techniques. The neural machine translation approach usually first encodes a variable-length source sentence, and then decodes it to a variable-length target sentence. Cho et al. (2014) and Sutskever et al. (2014) used two RNNs to encode a sentence to a fix-length vector and then decode the vector into a target sentence. Bahdanau et al. (2015) introduced the soft-attention technique to learn to jointly align and translate.

He et al. (2016a) proposed dual learning mechanism to tackle the data hunger issue in machine translation, inspired by the observation that the information feedback between the primal, translation from language A to language B, and the dual, translation from B to A, can help improve both translation models, with a policy gradient method, using the language model likelihood as the reward signal. Experiments showed that, with only 10% bilingual data for warm start and monolingual data, the dual learning approach performed comparably with previous neural machine translation methods with full bilingual data in English to French tasks. The dual learning mechanism may have extensions to many tasks, if the task has a dual form, e.g., speech recognition and text to speech, image caption and image generation, question answering and question generation, search and keyword extraction, etc.

See Wu et al. (2016); Johnson et al. (2016) for Google’s Neural Machine Translation System; Gehring et al. (2017) for convolutional sequence to sequence learning for fast neural machine translation; Klein et al. (2017) for OpenNMT, an open source neural machine translation system; Cheng et al. (2016) for semi-supervised learning for neural machine translation, and Wu et al. (2017) for adversarial neural machine translation. See Vaswani et al. (2017) for a new approach for translation that replaces CNN and RNN with attention and positional encoding. See Zhang et al. (2017b) for an open source toolkit for neural machine translation. See Monroe (2017) for a gentle introduction to translation.

5.3.3 TEXT GENERATION

Text generation models are usually based on n -gram, feed-forward neural networks, or recurrent neural networks, trained to predict next word given the previous ground truth words as inputs; then in testing, the trained models are used to generate a sequence word by word, using the generated words as inputs. The errors will accumulate on the way, causing the exposure bias issue. Moreover, these models are trained with word level losses, e.g., cross entropy, to maximize the probability of next word; however, the models are evaluated on a different metrics like BLEU.

Ranzato et al. (2016) proposed Mixed Incremental Cross-Entropy Reinforce (MIXER) for sequence prediction, with incremental learning and a loss function combining both REINFORCE and cross-entropy. MIXER is a sequence level training algorithm, aligning training and testing objective, such as BLEU, rather than predicting the next word as in previous works.

Bahdanau et al. (2017) proposed an actor-critic algorithm for sequence prediction, attempting to further improve Ranzato et al. (2016). The authors utilized a critic network to predict the value of a token, i.e., the expected score following the sequence prediction policy, defined by an actor network, trained by the predicted value of tokens. Some techniques are deployed to improve performance: SARSA rather than Monte-Carlo method to lessen the variance in estimating value functions; target network for stability; sampling prediction from a delayed actor whose weights are updated more slowly than the actor to be trained, to avoid the feedback loop when actor and critic need to be trained based on the output of each other; reward shaping to avoid the issue of sparse training signal.

Yu et al. (2017) proposed SeqGAN, sequence generative adversarial nets with policy gradient, integrating the adversarial scheme in Goodfellow et al. (2014). Li et al. (2017a) proposed to improve sequence generation by considering the knowledge about the future.

5.4 COMPUTER VISION

Computer vision is about how computers gain understanding from digital images or videos.

Mnih et al. (2014) introduced the recurrent attention model (RAM) to focus on selected sequence of regions or locations from an image or video for image classification and object detection. The authors used RL methods, in particular, REINFORCE algorithm, to train the model, to overcome the issue that the model is non-differentiable, and experimented on an image classification task and a dynamic visual control problem. We discuss attention in Section 4.1.

Some are integrating computer vision with natural language processing. Xu et al. (2015) integrated attention to image captioning, trained the hard version attention with REINFORCE, and showed the effectiveness of attention on Flickr8k, Flickr30k, and MS COCO datasets. See also Liu et al. (2016) and Lu et al. (2016) for image captioning. Strub et al. (2017) proposed end-to-end optimization with deep RL for goal-driven and visually grounded dialogue systems for GuessWhat?! game. Das et al. (2017) proposed to learn cooperative Visual Dialog agents with deep RL.

5.5 NEURAL ARCHITECTURE DESIGN

Neural networks architecture design is a notorious, nontrivial engineering issue. Neural architecture search provides a promising avenue to explore.

Zoph and Le (2017) proposed the neural architecture search to generate neural networks architectures with an RNN trained by RL, in particular, REINFORCE, searching from scratch in variable-length architecture space, to maximize the expected accuracy of the generated architectures on a validation set. In the RL formulation, a controller generates hyperparameters as a sequence of tokens, which are actions chosen from hyperparameters spaces; each gradient update to the policy parameters corresponds to training one generated network to convergence; an accuracy on a validation set is the reward signal. The neural architecture search can generate convolutional layers, with skip connections or branching layers, and recurrent cell architecture. The authors designed a parameter server approach to speed up training. Comparing with state-of-the-art methods, the proposed approach achieved competitive results for an image classification task with CIFAR-10 dataset; and better results for a language modeling task with Penn Treebank.

Baker et al. (2017) proposed a meta-learning approach, using Q-learning with ϵ -greedy exploration and experience replay, to generate CNN architectures automatically for a given learning task.

There are recent works exploring new neural architectures. Kaiser et al. (2017a) proposed to train a single model, MultiModel, which is composed of convolutional layers, an attention mechanism, and sparsely-gated layers, to learn multiple tasks from various domains, including image classification, image captioning and machine translation. Vaswani et al. (2017) proposed a new architecture for translation that replaces CNN and RNN with attention and positional encoding. Wang et al. (2016c) proposed the dueling network architecture to estimate state value function and associated advantage function, to combine them to estimate action value function for faster convergence. Tamar et al.

(2016) introduced Value Iteration Networks, a fully differentiable CNN planning module to approximate the value iteration algorithm, to learn to plan. Silver et al. (2016b) proposed the predictron to integrate learning and planning into one end-to-end training procedure with raw input in Markov reward process.

5.6 BUSINESS MANAGEMENT

Reinforcement learning has many applications in business management, like ads, recommendation, customer management, and marketing.

Li et al. (2010) formulated personalized news articles recommendation as a contextual bandit problem, to learn an algorithm to select articles sequentially for users based on contextual information of the user and articles, such as historical activities of the user and descriptive information and categories of content, and to take user-click feedback to adapt article selection policy to maximize total user clicks in the long run.

Theocharous et al. (2015) formulated a personalized Ad recommendation systems as a RL problem to maximize life-time value (LTV) with theoretical guarantees. This is in contrast to a myopic solution with supervised learning or contextual bandit formulation, usually with the performance metric of click through rate (CTR). As the models are hard to learn, the authors deployed a model-free approach to compute a lower-bound on the expected return of a policy to address the off-policy evaluation problem, i.e., how to evaluate a RL policy without deployment.

Li et al. (2015) also attempted to maximize lifetime value of customers. Silver et al. (2013) proposed concurrent reinforcement learning for the customer interaction problem. See Sutton and Barto (2017) for a detailed and intuitive description of some topics discussed here under the section title of personalized web services.

5.7 FINANCE

RL is a natural solution to some finance and economics problems (Hull, 2014; Luenberger, 1997), like option pricing (Longstaff and Schwartz, 2001; Tsitsiklis and Van Roy, 2001; Li et al., 2009), and multi-period portfolio optimization (Brandt et al., 2005; Neuneier, 1997), where value function based RL methods were used. Moody and Saffell (2001) proposed to utilize policy gradient to learn to trade; Deng et al. (2016) extended it with deep neural networks. Deep (reinforcement) learning would provide better solutions in some issues in risk management (Hull, 2014; Yu et al., 2009). The market efficiency hypothesis is fundamental in finance. However, there are well-known behavioral biases in human decision-making under uncertainty, in particular, prospect theory (Prashanth et al., 2016). A reconciliation is the adaptive markets hypothesis (Lo, 2004), which may be approached by reinforcement learning.

It is nontrivial for finance and economics academia to accept blackbox methods like neural networks; Heaton et al. (2016) may be regarded as an exception. However, there is a lecture in AFA 2017 annual meeting: Machine Learning and Prediction in Economics and Finance. We may also be aware that financial firms would probably hold state-of-the-art research/application results.

FinTech has been attracting attention, especially after the notion of big data. FinTech employs machine learning techniques to deal with issues like fraud detection (Phua et al., 2010), consumer credit risk (Khandani et al., 2010), etc.

5.8 HEALTHCARE

There are many opportunities and challenges in healthcare for machine learning (Miotto et al., 2017; Saria, 2014). Personalized medicine is getting popular in healthcare. It systematically optimizes the patient's health care, in particular, for chronic conditions and cancers using individual patient information, potentially from electronic health/medical record (EHR/EMR). Dynamic treatment regimes (DTRs) or adaptive treatment strategies are sequential decision making problems. Some issues in DTRs are not in standard RL. Shortreed et al. (2011) tackled the missing data problem, and designed methods to quantify the evidence of the learned optimal policy. Goldberg and Kosorok (2012) proposed methods for censored data (patients may drop out during the trial) and flexible number of stages. See Chakraborty and Murphy (2014) for a recent survey, and Kosorok and Moodie (2015)

for an edited book about recent progress in DTRs. Currently Q-learning is the RL method in DTRs. It is interesting to see the applications of deep RL methods in this field.

Some recent workshops at the intersection of machine learning and healthcare are: NIPS 2016 Workshop on Machine Learning for Health (<http://www.nipsml4hc.ws>) and NIPS 2015 Workshop on Machine Learning in Healthcare (<https://sites.google.com/site/nipsmlhc15/>).

5.9 INDUSTRY 4.0

The era of Industry 4.0 is approaching, e.g., see O'Donovan et al. (2015), and Preuveneers and Ilie-Zudor (2017). Reinforcement learning in particular, artificial intelligence in general, will be critical enabling techniques for many aspects of Industry 4.0, e.g., predictive maintenance, real-time diagnostics, and management of manufacturing activities and processes. Robots will prevail in Industry 4.0, and we discuss robotics in Section 5.2.

Surana et al. (2016) proposed to apply guided policy search (Levine et al., 2016a) as discussed in Section 5.2.1 to optimize trajectory policy of cold spray nozzle dynamics, to handle complex trajectories traversing by robotic agents. The authors generated cold spray surface simulation profiles to train the model.

5.10 SMART GRID

A smart grid is a power grid utilizing modern information technologies to create an intelligent electricity delivery network for electricity generation, transmission, distribution, consumption, and control (Fang et al., 2012). An important aspect is adaptive control (Anderson et al., 2011). Glavic et al. (2017) reviewed application of RL for electric power system decision and control. Here we briefly discuss demand response (Wen et al., 2015b; Ruelens et al., 2016).

Demand response systems motivate users to dynamically adapt electrical demands in response to changes in grid signals, like electricity price, temperature, and weather, etc. With suitable electricity prices, load of peak consumption may be rescheduled/lessened, to improve efficiency, reduce costs, and reduce risks. Wen et al. (2015b) proposed to design a fully automated energy management system with model-free reinforcement learning, so that it doesn't need to specify a disutility function to model users' dissatisfaction with job rescheduling. The authors decomposed the RL formulation over devices, so that the computational complexity grows linearly with the number of devices, and conducted simulations using Q-learning. Ruelens et al. (2016) tackled the demand response problem with batch RL. Wen et al. (2015b) took the exogenous prices as states, and Ruelens et al. (2016) utilized the average as feature extractor to construct states.

5.11 INTELLIGENT TRANSPORTATION SYSTEMS

Intelligent transportation systems apply advanced information technologies for tackling issues in transport networks, like congestion, safety, efficiency, etc., to make transport networks, vehicles and users smart.

An important issue in intelligent transportation systems is adaptive traffic signal control. El-Tantawy et al. (2013) proposed to model the adaptive traffic signal control problem as a multiple player stochastic game, and solve it with the approach of multi-agent RL (Shoham et al., 2007; Busoniu et al., 2008). Multi-agent RL integrates single agent RL with game theory, facing challenges of stability, nonstationarity, and curse of dimensionality. El-Tantawy et al. (2013) approached the issue of coordination by considering agents at neighbouring intersections. The authors validated their proposed approach with simulations, and real traffic data from the City of Toronto. El-Tantawy et al. (2013) didn't explore function approximation. See also van der Pol and Oliehoek (2017) for a recent work, and Mannion et al. (2016) for an experimental review, about applying RL to adaptive traffic signal control.

Self-driving vehicle is also a topic of intelligent transportation systems. See Bojarski et al. (2016), and Bojarski et al. (2017).

See NIPS 2016 Workshop on Machine Learning for Intelligent Transportation Systems. Check for a special issue of IEEE Transactions on Neural Networks and Learning Systems on Deep Reinforcement Learning and Adaptive Dynamic Programming, tentative publication date December 2017.

5.12 COMPUTER SYSTEMS

Computer systems are indispensable in our daily life and work, e.g., mobile phones, computers, and cloud computing. Control and optimization problems abound in computer systems, e.g., Mestres et al. (2016) proposed knowledge-defined networks, Gavrilovska et al. (2013) reviewed learning and reasoning techniques in cognitive radio networks, and Haykin (2005) discussed issues in cognitive radio, like channel state prediction and resource allocation. We also note that Internet of Things (IoT)(Xu et al., 2014) play an important role in Industry 4.0 as discussed in Section 5.9, in Smart Grid as discussed in Section 5.10, and in Intelligent Transportation Systems as discussed in Section 5.11.

Mao et al. (2016) studied resource management in systems and networking with deep RL. The authors proposed to tackle multi-resource cluster scheduling with policy gradient, in an online manner with dynamic job arrivals, optimizing various objectives like average job slowdown or completion time. The authors validated their proposed approach with simulation.

Mirhoseini et al. (2017) proposed to optimize device placement for Tensorflow computational graphs with RL. The authors deployed a sequence-to-sequence model to predict how to place subsets of operations in a Tensorflow graph on available devices, using the execution time of the predicted placement as reward signal for REINFORCE algorithm. The proposed method found placements of Tensorflow operations on devices for Inception-V3, recurrent neural language model and neural machine translation, yielding shorter execution time than those placements designed by human experts. Vinyals et al. (2015) and Bello et al. (2016) also discussed combinatorial optimization problems. Computation burden is one concern for a RL approach to search directly in the solution space of a combinatorial problem.

Liu et al. (2017) proposed a hierarchical framework to tackle resource allocation and power management in cloud computing with deep RL. The authors decomposed the problem as a global tier for virtual machines resource allocation and a local tier for servers power management. The authors validated their proposed approach with actual Google cluster traces. Such hierarchical framework/decomposition approach was to reduce state/action space, and to enable distributed operation of power management.

Google deployed machine learning for data centre power management, reducing energy consumption by 40%, <https://deepmind.com/blog/deepmind-ai-reduces-google-data-centre-cooling-bill-40/>. Optimizing memory control is discussed in Sutton and Barto (2017).

6 MORE TOPICS

We list more interesting and/or important topics we have not discussed in this overview as below, hoping it would provide pointers for those who may be interested in studying them further. Some topics/papers may not contain RL yet. However, we believe these are interesting and/or important directions for RL in the sense of either theory or application. It would be definitely more desirable if we could finish reviewing these, however, we leave it as future work.

- understanding deep learning, Daniely et al. (2016); Li et al. (2016b); Karpathy et al. (2016); Neyshabur et al. (2017); Shalev-Shwartz et al. (2017); Shwartz-Ziv and Tishby (2017); Zhang et al. (2017)
- interpretability, e.g., Al-Shedivat et al. (2017); Doshi-Velez and Kim (2017); Harrison et al. (2017); Lei et al. (2016); Lipton (2016); Miller (2017); Ribeiro et al. (2016); Huk Park et al. (2016), NIPS 2016 Workshop on Interpretable ML for Complex Systems, ICML Workshop on Human Interpretability in Machine Learning 2017, 2016
- usable machine learning, Bailis et al. (2017)
- expressivity, Raghu et al. (2016)
- testing, Pei et al. (2017)

-
- deep learning efficiency, e.g., Han et al. (2016), Spring and Shrivastava (2017), Sze et al. (2017)
 - optimization, e.g., Wilson et al. (2017), Czarnecki et al. (2017)
 - normalization, Klambauer et al. (2017), van Hasselt et al. (2016b)
 - hyperparameter learning, e.g. Andrychowicz et al. (2016)
 - curriculum learning, Andrychowicz et al. (2017), Graves et al. (2017), Held et al. (2017), Matisen et al. (2017)
 - professor forcing, Lamb et al. (2016)
 - new Q-value operators, Kavosh and Littman (2017), Haarnoja et al. (2017)
 - large action space, e.g., Dulac-Arnold et al. (2016); He et al. (2016c)
 - robust RL, e.g., Pinto et al. (2017)
 - multi-agent RL, e.g., Leibo et al. (2017); Foerster et al. (2016); Foerster et al. (2017); Foerster et al. (2017); Lowe et al. (2017); Omidshafiei et al. (2017); Sukhbaatar et al. (2016)
 - Bayesian RL (Ghavamzadeh et al., 2015)
 - neural episodic control, Pritzel et al. (2017)
 - continual learning, Kirkpatrick et al. (2017); Lopez-Paz and Ranzato (2017)
 - adversarial attacks, e.g., Huang et al. (2017); Madry et al. (2017); Papernot et al. (2016)
 - symbolic learning, Garnelo et al. (2016); Liang et al. (2017); Parisotto et al. (2017)
 - pathNet, Fernando et al. (2017)
 - evolution strategies, Salimans et al. (2017)
 - DeepForest, Zhou and Feng (2017)
 - deep probabilistic programming, Tran et al. (2017)
 - deep learning games, Schuurmans and Zinkevich (2016)
 - combinatorial optimization, e.g., Vinyals et al. (2015), Bello et al. (2016)
 - program learning, e.g., Balog et al. (2017); Cai et al. (2017); Denil et al. (2017); Parisotto et al. (2017); Reed and de Freitas (2016)
 - relational reasoning, e.g., Santoro et al. (2017), Watters et al. (2017)
 - proving, e.g., Loos et al. (2017); Rocktäschel and Riedel (2017)
 - music generation, e.g., Jaques et al. (2017)
 - retrosynthesis, e.g., Segler et al. (2017)
 - physics experiments, e.g., Denil et al. (2017)
 - quantum RL, e.g., Crawford et al. (2016), NIPS 2015 Workshop on Quantum Machine Learning

7 RESOURCES

We list a collection of deep RL resources including books, surveys, reports, online courses, tutorials, conferences, journals and workshops, blogs, testbed, and open source algorithm implementations. This by no means is complete.

It is essential to have a good understanding of reinforcement learning, before having a good understanding of deep reinforcement learning. We recommend to start with the textbook by Sutton and Barto (Sutton and Barto, 2017), the RL courses by Rich Sutton and by David Silver as the first two items in the Courses subsection below.

In the current information/social media age, we are overwhelmed by information, e.g., from Twitter, arXiv, Google+, etc. The skill to efficiently select the best information becomes essential. The Wild Week in AI (<http://www.wildml.com>) is an excellent series of weekly summary blogs. In an era of AI, we expect to see an AI agent to do such tasks like intelligently searching and summarizing relevant news, blogs, research papers, etc.

7.1 BOOKS

- the definitive and intuitive reinforcement learning book by Richard S. Sutton and Andrew G. Barto (Sutton and Barto, 2017)
- deep learning book by Ian Goodfellow, Yoshua Bengio, and Aaron Courville (Goodfellow et al., 2016)

7.2 MORE BOOKS

- theoretical RL books (Bertsekas, 2012; Bertsekas and Tsitsiklis, 1996; Szepesvári, 2010)
- an operations research oriented RL book (Powell, 2011)
- an edited RL book (Wiering and van Otterlo, 2012)
- Markov decision processes (Puterman, 2005)
- deep learning (Deng and Dong, 2014)
- machine learning (Bishop, 2011; Hastie et al., 2009; Haykin, 2008; James et al., 2013; Kuhn and Johnson, 2013; Murphy, 2012; Provost and Fawcett, 2013; Zhou, 2016)
- artificial intelligence (Russell and Norvig, 2009)
- natural language processing (NLP) (Goldberg, 2017; Deng and Liu, 2017)
- semi-supervised learning (Zhu and Goldberg, 2009)
- game theory (Leyton-Brown and Shoham, 2008)

7.3 SURVEYS AND REPORTS

- reinforcement learning (Littman, 2015; Kaelbling et al., 1996; Grondman et al., 2012)
- deep learning (LeCun et al., 2015; Schmidhuber, 2015; Bengio, 2009; Wang and Raj, 2017)
- efficient processing of deep neural networks (Sze et al., 2017)
- machine learning (Jordan and Mitchell, 2015)
- practical machine learning advices (Domingos, 2012; Smith, 2017; Zinkevich, 2017)
- natural language processing (NLP) (Hirschberg and Manning, 2015; Cho, 2015)
- spoken dialogue systems (Deng and Li, 2013; Hinton et al., 2012; He and Deng, 2013; Young et al., 2013)
- robotics (Kober et al., 2013)
- transfer learning (Taylor and Stone, 2009; Pan and Yang, 2010; Weiss et al., 2016)
- Bayesian RL (Ghavamzadeh et al., 2015)
- AI safety (Amodei et al., 2016; García and Fernández, 2015)
- Monte Carlo tree search (MCTS) (Browne et al., 2012; Gelly et al., 2012)
- multi-agent RL (Shoham et al., 2007; Busoniu et al., 2008)

7.4 COURSES

- Richard Sutton, Reinforcement Learning, 2016, slides, assignments, reading materials, etc. <http://www.incompleteideas.net/sutton/609%20dropbox/>
- David Silver, Reinforcement Learning, 2015, slides (goo.gl/UqaxlO), video-lectures (goo.gl/7BVRkT)
- Sergey Levine, John Schulman and Chelsea Finn, CS 294: Deep Reinforcement Learning, Spring 2017, <http://rll.berkeley.edu/deeprlcourse/>
- Katerina Fragkiadaki, Ruslan Satakhutdinov, Deep Reinforcement Learning and Control, Spring 2017, <https://katefvision.github.io>
- Charles Isbell, Michael Littman and Pushkar Kolhe, Udacity: Machine Learning: Reinforcement Learning, goo.gl/eyvLfg

-
- Nando de Freitas, Deep Learning Lectures, <https://www.youtube.com/user/ProfNandoDF>
 - Fei-Fei Li, Andrej Karpathy and Justin Johnson, CS231n: Convolutional Neural Networks for Visual Recognition, <http://cs231n.stanford.edu>
 - Richard Socher, CS224d: Deep Learning for Natural Language Processing, <http://cs224d.stanford.edu>
 - Brendan Shillingford, Yannis Assael, Chris Dyer, Oxford Deep NLP 2017 course, <https://github.com/oxford-cs-deepnlp-2017>
 - Pieter Abbeel, Advanced Robotics, Fall 2015, <https://people.eecs.berkeley.edu/~pabbeel/cs287-fa15/>
 - Emo Todorov, Intelligent control through learning and optimization, <http://homes.cs.washington.edu/~todorov/courses/amath579/index.html>
 - Abdeslam Boularias, Robot Learning Seminar, <http://www.abdeslam.net/robotlearningseminar>
 - MIT 6.S094: Deep Learning for Self-Driving Cars, <http://selfdrivingcars.mit.edu>
 - Jeremy Howard, Practical Deep Learning For Coders, <http://course.fast.ai>

7.5 TUTORIALS

- Rich Sutton, Introduction to Reinforcement Learning with Function Approximation, <https://www.microsoft.com/en-us/research/video/tutorial-introduction-to-reinforcement-learning-with-function-approximation/>
- Deep Reinforcement Learning
 - David Silver, ICML 2016
 - David Silver, 2nd Multidisciplinary Conference on Reinforcement Learning and Decision Making (RLDM), Edmonton, Alberta, Canada, 2015; http://videlectures.net/rldm2015_silver_reinforcement_learning/
 - John Schulman, Deep Learning School, 2016
 - Pieter Abbeel, Deep Learning Summer School, 2016; http://videlectures.net/deeplearning2016_abbeel_deep_reinforcement/
 - Pieter Abbeel and John Schulman, Deep Reinforcement Learning Through Policy Optimization, NIPS 2016
 - Sergey Levine and Chelsea Finn, Deep Reinforcement Learning, Decision Making, and Control, ICML 2017
- John Schulman, The Nuts and Bolts of Deep Reinforcement Learning Research, Deep Reinforcement Learning Workshop, NIPS 2016
- Joelle Pineau, Introduction to Reinforcement Learning, Deep Learning Summer School, 2016; http://videlectures.net/deeplearning2016_pineau_reinforcement_learning/
- Andrew Ng, Nuts and Bolts of Building Applications using Deep Learning, NIPS 2016
- Deep Learning Summer School, 2016, 2015
- Deep Learning and Reinforcement Learning Summer Schools, 2017
- Simons Institute Interactive Learning Workshop, 2017
- Simons Institute Representation Learning Workshop, 2017
- Simons Institute Computational Challenges in Machine Learning Workshop, 2017

7.6 CONFERENCES, JOURNALS AND WORKSHOPS

- NIPS: Neural Information Processing Systems
- ICML: International Conference on Machine Learning
- ICLR: International Conference on Learning Representation
- RLDM: Multidisciplinary Conference on Reinforcement Learning and Decision Making
- EWRL: European Workshop on Reinforcement Learning

- AAAI, IJCAI, ACL, EMNLP, SIGDIAL, ICRA, IROS, KDD, SIGIR, CVPR, etc.
- Science Robotics, JMLR, MLJ, AIJ, JAIR, PAMI, etc
- Nature May 2015, Science July 2015, survey papers on machine learning/AI
- Science, July 7, 2017 issue, The Cyberscientist, a special issue about AI
- Deep Reinforcement Learning Workshop, NIPS 2016, 2015; IJCAI 2016
- Deep Learning Workshop, ICML 2016
- <http://distill.pub>

7.7 BLOGS

- Denny Britz, The Wild Week in AI, a weekly AI & deep learning newsletter, www.wildml.com, esp. goo.gl/MyrwDC
- Andrej Karpathy, karpathy.github.io, esp. goo.gl/1hkKrb
- Junling Hu, Reinforcement learning explained - learning to act based on long-term payoffs <https://www.oreilly.com/ideas/reinforcement-learning-explained>
- Li Deng, How deep reinforcement learning can help chatbots <https://venturebeat.com/2016/08/01/how-deep-reinforcement-learning-can-help-chatbots/>
- Christopher Olah, colah.github.io
- Reinforcement Learning, <https://www.technologyreview.com/s/603501/10-breakthrough-technologies-2017-reinforcement-learning/>
- Deep Learning, <https://www.technologyreview.com/s/513696/deep-learning/>
- Berkeley AI Research Blog, <http://bair.berkeley.edu/blog/>

7.8 TESTBEDS

- The Arcade Learning Environment (ALE) (Bellemare et al., 2013) is a framework composed of Atari 2600 games to develop and evaluate AI agents.
- DeepMind released a first-person 3D game platform DeepMind Lab (Beattie et al., 2016). Deepmind and Blizzard will collaborate to release the Starcraft II AI research environment (goo.gl/Ptiwfg).
- OpenAI Gym (<https://gym.openai.com>) is a toolkit for the development of RL algorithms, consisting of environments, e.g., Atari games and simulated robots, and a site for the comparison and reproduction of results.
- OpenAI Universe (<https://universe.openai.com>) is used to turn any program into a Gym environment. Universe has already integrated many environments, including Atari games, flash games, browser tasks like Mini World of Bits and real-world browser tasks. Recently, GTA V was added to Universe for self-driving vehicle simulation.
- FAIR TorchCraft (Synnaeve et al., 2016) is a library for Real-Time Strategy (RTS) games such as StarCraft: Brood War.
- ParlAI is a framework for dialogue research, implemented in Python, open-sourced by Facebook. <https://github.com/facebookresearch/ParlAI>
- ELF, an extensive, lightweight and flexible platform for RL research (Tian et al., 2017)
- Project Malmö (<https://github.com/Microsoft/malmo>), from Microsoft, is an AI research and experimentation platform built on top of Minecraft.
- Twitter open-sourced torch-twrl, a framework for RL development.
- ViZDoom is a Doom-based AI research platform for visual RL (Kempka et al., 2016).
- Baidu Apollo Project, self-driving open-source, <http://apollo.auto>
- TORCS is a car racing simulator (Bernhard Wymann et al., 2014).
- MuJoCo, Multi-Joint dynamics with Contact, is a physics engine, <http://www.mujoco.org>.
- Nogueira and Cho (2016) presented WebNav Challenge for Wikipedia links navigation.
- RLGlue (Tanner and White, 2009) is a language-independent software for RL experiments. It may need extensions to accommodate progress in deep learning.

7.9 ALGORITHM IMPLEMENTATIONS

We collect implementations of algorithms, either classical ones as in a textbook like Sutton and Barto (2017) or in recent papers.

- Learning Reinforcement Learning (with Code, Exercises and Solutions), to accompany Richard Sutton’s RL book and David Silver’s RL course, <http://www.wildml.com/2016/10/learning-reinforcement-learning/>
- OpenAI Baselines: high-quality implementations of reinforcement learning algorithms, <https://github.com/openai/baselines>
- TensorFlow implementation of Deep Reinforcement Learning papers, <https://github.com/carpedm20/deep-rl-tensorflow>
- Deep reinforcement learning for Keras, <https://github.com/matthiasplappert/keras-rl>
- Code Implementations for NIPS 2016 papers, <http://bit.ly/2hSaOyx>
- Benchmark results of various policy optimization algorithms (Duan et al., 2016), <https://github.com/rllab/rllab>
- Tensor2Tensor (T2T) (Vaswani et al., 2017; Kaiser et al., 2017a;b)
- DQN (Mnih et al., 2015), <https://sites.google.com/a/deepmind.com/dqn/>
- Tensorflow implementation of DQN (Mnih et al., 2015), <https://github.com/devsisters/DQN-tensorflow>
- Deep Q Learning with Keras and Gym, <https://keon.io/deep-q-learning/>
- Deep Exploration via Bootstrapped DQN (Osband et al., 2016), a Torch implementation, <https://github.com/iassael/torch-bootstrapped-dqn>
- DarkForest, the Facebook Go engine (Github), <https://github.com/facebookresearch/darkforestGo>
- Using Keras and Deep Q-Network to Play FlappyBird, <https://yanpanlau.github.io/2016/07/10/FlappyBird-Keras.html>
- Deep Deterministic Policy Gradients (Lillicrap et al., 2016) in TensorFlow, <http://pemami4911.github.io/blog/2016/08/21/ddpg-rl.html>
- Deep Deterministic Policy Gradient (Lillicrap et al., 2016) to play TORCS, <https://yanpanlau.github.io/2016/10/11/Torcs-Keras.html>
- Reinforcement learning with unsupervised auxiliary tasks (Jaderberg et al., 2017), <https://github.com/miyosuda/unreal>
- Learning to communicate with deep multi-agent reinforcement learning, <https://github.com/iassael/learning-to-communicate>
- Deep Reinforcement Learning: Playing a Racing Game - Byte Tank, <http://bit.ly/2pVIP4i>
- TensorFlow implementation of DeepMind’s Differential Neural Computers (Graves et al., 2016), <https://github.com/Mostafa-Samir/DNC-tensorflow>
- Learning to Learn (Reed and de Freitas, 2016) in TensorFlow, <https://github.com/deepmind/learning-to-learn>
- Value Iteration Networks (Tamar et al., 2016) in Tensorflow, <https://github.com/TheAbhiKumar/tensorflow-value-iteration-networks>
- Tensorflow implementation of the Predictron (Silver et al., 2016b), <https://github.com/zhongwen/predictron>
- Meta Reinforcement Learning (Wang et al., 2016a) in Tensorflow, <https://github.com/awjuliani/Meta-RL>
- Generative adversarial imitation learning (Ho and Ermon, 2016), containing an implementation of Trust Region Policy Optimization (TRPO) (Schulman et al., 2015), <https://github.com/openai/imitation>
- Starter code for evolution strategies (Salimans et al., 2017), <https://github.com/openai/evolution-strategies-starter>
- Transfer learning (Long et al., 2015; 2016), <https://github.com/thuml/transfer-caffe>
- DeepForest (Zhou and Feng, 2017), <http://lamda.nju.edu.cn/files/gcforest.zip>

8 BRIEF SUMMARY

We list some RL issues and corresponding proposed approaches covered in this overview, as well as some classical work. One direction of future work is to further refine this section, especially for issues and solutions in applications.

- issue: prediction, policy evaluation
proposed approaches:
 - temporal difference (TD) learning (Sutton, 1988)
- issue: control, finding optimal policy (classical work)
proposed approaches:
 - Q-learning (Watkins and Dayan, 1992)
 - policy gradient (Williams, 1992)
 - ◊ reduce variance of gradient estimate: baseline, advantage function (Williams, 1992; Sutton et al., 2000)
 - actor-critic (Barto et al., 1983)
 - SARSA (Sutton and Barto, 2017)
- issue: the deadly triad: instability and divergence when combining off-policy, function approximation, and bootstrapping
proposed approaches:
 - DQN with experience replay (Lin, 1992) and target network (Mnih et al., 2015)
 - ◊ overestimate problem in Q-learning: double DQN (van Hasselt et al., 2016a)
 - ◊ prioritized experience replay (Schaul et al., 2016)
 - ◊ better exploration strategy (Osband et al., 2016)
 - ◊ optimality tightening to accelerate DQN (He et al., 2017a)
 - ◊ reduce variability and instability with averaged-DQN (Anschel et al., 2017)
 - dueling architecture (Wang et al., 2016c)
 - asynchronous methods (Mnih et al., 2016)
 - trust region policy optimization (Schulman et al., 2015)
 - distributed proximal policy optimization (Heess et al., 2017)
 - combine policy gradient and Q-learning (O’Donoghue et al., 2017; Nachum et al., 2017; Gu et al., 2017; Schulman et al., 2017)
 - GTD (Sutton et al., 2009a;b; Mahmood et al., 2014)
 - Emphatic-TD (Sutton et al., 2016)
- issue: train perception and control jointly end-to-end
proposed approaches:
 - guided policy search (Levine et al., 2016a)
- issue: data/sample efficiency
proposed approaches:
 - Q-learning, actor-critic
 - actor-critic with experience replay (Wang et al., 2017b)
 - PGQ, policy gradient and Q-learning (O’Donoghue et al., 2017)
 - Q-Prop, policy gradient with off-policy critic (Gu et al., 2017)
 - return-based off-policy control, Retrace (Munos et al., 2016), Reactor (Gruslys et al., 2017)
 - learning to learn, e.g., Duan et al. (2017); Wang et al. (2016a); Lake et al. (2015)
- issue: reward function not available
proposed approaches:
 - imitation learning

-
- inverse RL (Ng and Russell, 2000)
 - learn from demonstration (Hester et al., 2017)
 - imitation learning with GANs (Ho and Ermon, 2016; Stadie et al., 2017)
 - train dialogue policy jointly with reward model (Su et al., 2016b)
 - issue: exploration-exploitation tradeoff
 - proposed approaches:
 - unify count-based exploration and intrinsic motivation (Bellemare et al., 2016)
 - under-appreciated reward exploration (Nachum et al., 2017)
 - deep exploration via bootstrapped DQN (Osband et al., 2016)
 - variational information maximizing exploration (Houthoofd et al., 2016)
 - issue: model-based learning
 - proposed approaches:
 - Dyna-Q (Sutton, 1990)
 - combine model-free and model-based RL (Chebotar et al., 2017)
 - issue: model-free planning
 - proposed approaches:
 - value iteration networks (Tamar et al., 2016)
 - predictron (Silver et al., 2016b)
 - issue: focus on salient parts
 - proposed approaches: attention
 - object detection (Mnih et al., 2014)
 - neural machine translation (Bahdanau et al., 2015)
 - image captioning (Xu et al., 2015)
 - replace CNN and RNN with attention in sequence modelling (Vaswani et al., 2017)
 - issue: data storage over long time, separating from computation
 - proposed approaches: memory
 - differentiable neural computer (DNC) with external memory (Graves et al., 2016)
 - issue: benefit from non-reward training signals in environments
 - proposed approaches: unsupervised Learning
 - Horde (Sutton et al., 2011)
 - unsupervised reinforcement and auxiliary learning (Jaderberg et al., 2017)
 - learn to navigate with unsupervised auxiliary learning (Mirowski et al., 2017)
 - generative adversarial networks (GANs) (Goodfellow et al., 2014)
 - issue: learn knowledge from different domains
 - proposed approaches: transfer Learning (Taylor and Stone, 2009; Pan and Yang, 2010; Weiss et al., 2016)
 - learn invariant features to transfer skills (Gupta et al., 2017)
 - issue: benefit from both labelled and unlabelled data
 - proposed approaches: semi-supervised learning (Zhu and Goldberg, 2009)
 - learn with MDPs both with and without reward functions (Finn et al., 2017)
 - learn with expert’s trajectories and those may not from experts (Audiffren et al., 2015)
 - issue: learn, plan, and represent knowledge with spatio-temporal abstraction at multiple levels
 - proposed approaches: hierarchical RL (Barto and Mahadevan, 2003)
 - strategic attentive writer to learn macro-actions (Vezhnevets et al., 2016)
 - integrate temporal abstraction with intrinsic motivation (Kulkarni et al., 2016)

-
- stochastic neural networks for hierarchical RL (Florensa et al., 2017)
 - lifelong learning with hierarchical RL (Tessler et al., 2017)
 - issue: adapt rapidly to new tasks
 - proposed approaches: learning to learn
 - learn a flexible RNN model to handle a family of RL tasks (Duan et al., 2017; Wang et al., 2016a)
 - one/few/zero-shot learning (Duan et al., 2017; Johnson et al., 2016; Kaiser et al., 2017b; Koch et al., 2015; Lake et al., 2015; Li and Malik, 2017; Ravi and Larochelle, 2017; Vinyals et al., 2016)
 - issue: gigantic search space
 - proposed approaches:
 - integrate supervised learning, reinforcement learning, and Monte-Carlo tree search as in AlphaGo (Silver et al., 2016a)
 - issue: neural networks architecture design
 - proposed approaches:
 - neural architecture search (Baker et al., 2017; Zoph and Le, 2017)
 - new architectures, e.g., Kaiser et al. (2017a), Silver et al. (2016b), Tamar et al. (2016), Vaswani et al. (2017), Wang et al. (2016c)

9 DISCUSSIONS

It is both the best and the worst of times for the field of deep RL, for the same reason: it has been growing so fast and so enormously. We have been witnessing breakthroughs, exciting new methods and applications, and we expect to see much more and much faster. As a consequence, this overview is incomplete, in the sense of both depth and width. However, we attempt to summarize important achievements and discuss potential directions and applications in this amazing field.

In this overview, we summarize six core elements – value function, policy, reward, model, planning, and exploration; six important mechanisms – attention and memory, unsupervised learning, transfer learning, semi-supervised learning, hierarchical RL, and learning to learn; and twelve applications – games, robotics, natural language processing, computer vision, neural architecture design, business management, finance, healthcare, Industry 4.0, smart grid, intelligent transportation systems, and computer systems. We also discuss background of machine learning, deep learning, and reinforcement learning, and list a collection of RL resources.

We have been witnessing breakthroughs, three papers about or using deep RL published in Nature in less than two years: deep Q-network (Mnih et al., 2015), AlphaGo (Silver et al., 2016a) and differentiable neural computer (Graves et al., 2016); We have already seen many extensions to, improvements for and applications of deep Q-network (Mnih et al., 2015). The mechanisms of attention and memory (Graves et al., 2016) has been attracting much attention.

Novel architectures and applications using deep RL were recognized in top tier conferences as best (student) papers in 2016: dueling network architectures (Wang et al., 2016c) at ICML, spoken dialogue systems (Su et al., 2016b) at ACL (student), information extraction (Narasimhan et al., 2016) at EMNLP, and value iteration networks (Tamar et al., 2016) at NIPS. Exciting achievements abound: asynchronous methods (Mnih et al., 2016), dual learning for machine translation (He et al., 2016a), guided policy search (Levine et al., 2016a), generative adversarial imitation learning (Ho and Ermon, 2016), unsupervised reinforcement and auxiliary learning (Jaderberg et al., 2017), and neural architecture design (Zoph and Le, 2017), etc.

Value function is central to reinforcement learning, e.g., in deep Q-network and its many extensions. Policy optimization approaches have been gaining traction, in many, diverse applications, e.g., robotics, neural architecture design, spoken dialogue systems, machine translation, attention, and learning to learn, and this list is boundless. New learning mechanisms have emerged, e.g., using transfer/unsupervised/semi-supervised learning to improve the quality and speed of learning, and more new mechanisms will be emerging. This is the renaissance of reinforcement learn-

ing (Krakovsky, 2016). In fact, reinforcement learning and deep learning have been making steady progress even during the last AI winter.

It is essential to consider issues of learning models, like stability, convergence, accuracy, data efficiency, scalability, speed, simplicity, interpretability, robustness, and safety, etc. It is important to investigate comments/criticisms, e.g., from cognitive science, like intuitive physics, intuitive psychology, causal model, compositionality, learning to learn, and act in real time (Lake et al., 2016), for stronger AI. See also Peter Norvig’s perspective at <http://bit.ly/2qpehcd>.

Nature in May 2015 and Science in July 2015 featured survey papers on machine learning/AI. Science Robotics launched in 2016. The July 7, 2017 issue of Science has a special issue about AI on The Cyberscientist. The coverage of AI by premier journals like Nature and Science and the launch of Science Robotics illustrate the apparent importance of AI.

It is worthwhile to envision deep RL considering perspectives of government, academia and industry on AI, e.g., Artificial Intelligence, Automation, and the economy, Executive Office of the President, USA; Artificial Intelligence and Life in 2030 - One Hundred Year Study on Artificial Intelligence: Report of the 2015-2016 Study Panel, Stanford University; and AI, Machine Learning and Data Fuel the Future of Productivity by The Goldman Sachs Group, Inc., etc. See also the recent AI Frontiers Conference, <https://www.aifrontiers.com>.

Deep learning was among MIT Technology Review 10 Breakthrough Technologies in 2013. We have been witnessing the dramatic development of deep learning in both academia and industry in the last few years. Reinforcement learning was among MIT Technology Review 10 Breakthrough Technologies in 2017. We will see both deep learning and reinforcement learning prospering in the coming years and beyond.

Deep learning, in this third wave of AI, will have deeper influences, as we have already seen from its many achievements. Reinforcement learning, as a more general learning and decision making paradigm, will deeply influence deep learning, machine learning, and artificial intelligence in general. Deepmind, conducting leading research in deep reinforcement learning, recently opened its first ever international AI research office in Alberta, Canada, co-locating with the major research center for reinforcement learning led by Rich Sutton. It is interesting to mention that when Professor Rich Sutton started working in the University of Alberta in 2003, he named his lab RLAI: Reinforcement Learning and Artificial Intelligence.

ACKNOWLEDGEMENT

We appreciate comments from Baochun Bai, Kan Deng, Hai Fang, Hua He, Junling Hu, Ruitong Huang, Lihong Li, Dale Schuurmans, David Silver, Rich Sutton, Csaba Szepesvári, Arash Tavakoli, Cameron Upright, Yi Wan, Qing Yu, Yaoliang Yu and attendants of various seminars and webinars. Any remaining issues and errors are our own.

REFERENCES

- Abbeel, P. and Ng, A. Y. (2004). Apprenticeship learning via inverse reinforcement learning. In *the International Conference on Machine Learning (ICML)*.
- Al-Shedivat, M., Dubey, A., and Xing, E. P. (2017). Contextual Explanation Networks. *ArXiv e-prints*.
- Amodei, D., Olah, C., Steinhardt, J., Christiano, P., Schulman, J., and Mané, D. (2016). Concrete Problems in AI Safety. *ArXiv e-prints*.
- Anderson, R. N., Boulanger, A., Powell, W. B., and Scott, W. (2011). Adaptive stochastic control for the smart grid. *Proceedings of the IEEE*, 99(6):1098–1115.
- Andreas, J., Klein, D., and Levine, S. (2017). Modular multitask reinforcement learning with policy sketches. In *the International Conference on Machine Learning (ICML)*.
- Andrychowicz, M., Denil, M., Colmenarejo, S. G., Hoffman, M. W., Pfau, D., Schaul, T., Shillingford, B., and de Freitas, N. (2016). Learning to learn by gradient descent by gradient descent. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.

-
- Andrychowicz, M., Wolski, F., Ray, A., Schneider, J., Fong, R., Welinder, P., McGrew, B., Tobin, J., Abbeel, P., and Zaremba, W. (2017). Hindsight Experience Replay. *ArXiv e-prints*.
- Anschel, O., Baram, N., and Shimkin, N. (2017). Averaged-DQN: Variance reduction and stabilization for deep reinforcement learning. In *the International Conference on Machine Learning (ICML)*.
- Argall, B. D., Chernova, S., Veloso, M., and Browning, B. (2009). A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57(5):469–483.
- Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein GAN. *ArXiv e-prints*.
- Asri, L. E., He, J., and Suleman, K. (2016). A sequence-to-sequence model for user simulation in spoken dialogue systems. In *Annual Meeting of the International Speech Communication Association (INTERSPEECH)*.
- Audiffren, J., Valko, M., Lazaric, A., and Ghavamzadeh, M. (2015). Maximum entropy semi-supervised inverse reinforcement learning. In *the International Joint Conference on Artificial Intelligence (IJCAI)*.
- Ba, J., Hinton, G. E., Mnih, V., Leibo, J. Z., and Ionescu, C. (2016). Using fast weights to attend to the recent past. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Ba, J., Mnih, V., and Kavukcuoglu, K. (2014). Multiple object recognition with visual attention. In *the International Conference on Learning Representations (ICLR)*.
- Babaeizadeh, M., Frosio, I., Tyree, S., Clemons, J., and Kautz, J. (2017). Reinforcement learning through asynchronous advantage actor-critic on a gpu. In *the International Conference on Learning Representations (ICLR)*.
- Bahdanau, D., Brakel, P., Xu, K., Goyal, A., Lowe, R., Pineau, J., Courville, A., and Bengio, Y. (2017). An actor-critic algorithm for sequence prediction. In *the International Conference on Learning Representations (ICLR)*.
- Bahdanau, D., Cho, K., and Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. In *the International Conference on Learning Representations (ICLR)*.
- Bailis, P., Olukoton, K., Re, C., and Zaharia, M. (2017). Infrastructure for Usable Machine Learning: The Stanford DAWN Project. *ArXiv e-prints*.
- Baird, L. (1995). Residual algorithms: Reinforcement learning with function approximation. In *the International Conference on Machine Learning (ICML)*.
- Baker, B., Gupta, O., Naik, N., and Raskar, R. (2017). Designing neural network architectures using reinforcement learning. In *the International Conference on Learning Representations (ICLR)*.
- Balog, M., Gaunt, A. L., Brockschmidt, M., Nowozin, S., and Tarlow, D. (2017). Deepcoder: Learning to write programs. In *the International Conference on Learning Representations (ICLR)*.
- Barto, A. G. and Mahadevan, S. (2003). Recent advances in hierarchical reinforcement learning. *Discrete Event Dynamic Systems*, 13(4):341–379.
- Barto, A. G., Sutton, R. S., and Anderson, C. W. (1983). Neuronlike elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics*, 13:835–846.
- Beattie, C., Leibo, J. Z., Teplyaev, D., Ward, T., Wainwright, M., Küttler, H., Lefrancq, A., Green, S., Valdés, V., Sadik, A., Schrittwieser, J., Anderson, K., York, S., Cant, M., Cain, A., Bolton, A., Gaffney, S., King, H., Hassabis, D., Legg, S., and Petersen, S. (2016). DeepMind Lab. *ArXiv e-prints*.
- Bellemare, M. G., Danihelka, I., Dabney, W., Mohamed, S., Lakshminarayanan, B., Hoyer, S., and Munos, R. (2017). The Cramer Distance as a Solution to Biased Wasserstein Gradients. *ArXiv e-prints*.

-
- Bellemare, M. G., Naddaf, Y., Veness, J., and Bowling, M. (2013). The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279.
- Bellemare, M. G., Schaul, T., Srinivasan, S., Saxton, D., Ostrovski, G., and Munos, R. (2016). Unifying count-based exploration and intrinsic motivation. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Bello, I., Pham, H., Le, Q. V., Norouzi, M., and Bengio, S. (2016). Neural Combinatorial Optimization with Reinforcement Learning. *ArXiv e-prints*.
- Bengio, Y. (2009). Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127.
- Bengio, Y., Louradour, J., Collobert, R., and Weston, J. (2009). Curriculum learning. In *the International Conference on Machine Learning (ICML)*.
- Bernhard Wymann, E. E., Guionneau, C., Dimitrakakis, C., and Rémi Coulom, A. S. (2014). TORCS, The Open Racing Car Simulator. "<http://www.torcs.org>".
- Berthelot, D., Schumm, T., and Metz, L. (2017). BEGAN: Boundary Equilibrium Generative Adversarial Networks. *ArXiv e-prints*.
- Bertsekas, D. P. (2012). *Dynamic programming and optimal control (Vol. II, 4th Edition: Approximate Dynamic Programming)*. Athena Scientific, Massachusetts, USA.
- Bertsekas, D. P. and Tsitsiklis, J. N. (1996). *Neuro-Dynamic Programming*. Athena Scientific.
- Bishop, C. (2011). *Pattern Recognition and Machine Learning*. Springer.
- Bojarski, M., Testa, D. D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L. D., Monfort, M., Muller, U., Zhang, J., Zhang, X., Zhao, J., and Zieba, K. (2016). End to End Learning for Self-Driving Cars. *ArXiv e-prints*.
- Bojarski, M., Yeres, P., Choromanska, A., Choromanski, K., Firner, B., Jackel, L., and Muller, U. (2017). Explaining How a Deep Neural Network Trained with End-to-End Learning Steers a Car. *ArXiv e-prints*.
- Bordes, A., Boureau, Y.-L., and Weston, J. (2017). Learning end-to-end goal-oriented dialog. In *the International Conference on Learning Representations (ICLR)*.
- Bowling, M., Burch, N., Johanson, M., and Tammelin, O. (2015). Heads-up limit hold'em poker is solved. *Science*, 347(6218):145–149.
- Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press.
- Bradtke, S. J. and Barto, A. G. (1996). Linear least-squares algorithms for temporal difference learning. *Machine Learning*, 22(1-3):33–57.
- Brandt, M. W., Goyal, A., Santa-Clara, P., and Stroud, J. R. (2005). A simulation approach to dynamic portfolio choice with an application to learning about return predictability. *The Review of Financial Studies*, 18(3):831–873.
- Browne, C., Powley, E., Whitehouse, D., Lucas, S., Cowling, P. I., Rohlfshagen, P., Tavener, S., Perez, D., Samothrakis, S., and Colton, S. (2012). A survey of Monte Carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in Games*, 4(1):1–43.
- Busoniu, L., Babuska, R., and Schutter, B. D. (2008). A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics - Part C: Applications and Reviews*, 38(2).
- Cai, J., Shin, R., and Song, D. (2017). Making neural programming architectures generalize via recursion. In *the International Conference on Learning Representations (ICLR)*.

-
- Carleo, G. and Troyer, M. (2017). Solving the quantum many-body problem with artificial neural networks. *Science*, 355(6325):602–606.
- Celikyilmaz, A., Deng, L., Li, L., and Wang, C. (2017). Scaffolding Networks: Incremental Learning and Teaching Through Questioning. *ArXiv e-prints*.
- Chakraborty, B. and Murphy, S. A. (2014). Dynamic treatment regimes. *Annual Review of Statistics and Its Application*, 1:447–464.
- Chebotar, Y., Hausman, K., Zhang, M., Sukhatme, G., Schaal, S., and Levine, S. (2017). Combining model-based and model-free updates for trajectory-centric reinforcement learning. In *the International Conference on Machine Learning (ICML)*.
- Chebotar, Y., Kalakrishnan, M., Yahya, A., Li, A., Schaal, S., and Levine, S. (2016). Path integral guided policy search. *ArXiv e-prints*.
- Chen, J., Huang, P.-S., He, X., Gao, J., and Deng, L. (2016). Unsupervised Learning of Predictors from Unpaired Input-Output Samples. *ArXiv e-prints*.
- Chen, Y.-N., Hakkani-Tur, D., Tur, G., Celikyilmaz, A., Gao, J., and Deng, L. (2016a). Knowledge as a Teacher: Knowledge-Guided Structural Attention Networks. *ArXiv e-prints*.
- Chen, Y.-N. V., Hakkani-Tür, D., Tur, G., Gao, J., and Deng, L. (2016b). End-to-end memory networks with knowledge carryover for multi-turn spoken language understanding. In *Annual Meeting of the International Speech Communication Association (INTERSPEECH)*.
- Chen, Z. and Yi, D. (2017). The Game Imitation: Deep Supervised Convolutional Networks for Quick Video Game AI. *ArXiv e-prints*.
- Cheng, Y., Xu, W., He, Z., He, W., Wu, H., Sun, M., and Liu, Y. (2016). Semi-supervised learning for neural machine translation. In *the Association for Computational Linguistics annual meeting (ACL)*.
- Cho, K. (2015). Natural Language Understanding with Distributed Representation. *ArXiv e-prints*.
- Cho, K., van Merriënboer, B., Gulcehre, C., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- Choi, E., Hewlett, D., Polosukhin, I., Lacoste, A., Uszkoreit, J., and Berant, J. (2017). Coarse-to-fine question answering for long documents. In *the Association for Computational Linguistics annual meeting (ACL)*.
- Christiano, P., Leike, J., Brown, T. B., Martic, M., Legg, S., and Amodei, D. (2017). Deep reinforcement learning from human preferences. *ArXiv e-prints*.
- Crawford, D., Levit, A., Ghadermarzy, N., Oberoi, J. S., and Ronagh, P. (2016). Reinforcement Learning Using Quantum Boltzmann Machines. *ArXiv e-prints*.
- Czarnecki, W. M., Świrszcz, G., Jaderberg, M., Osindero, S., Vinyals, O., and Kavukcuoglu, K. (2017). Understanding Synthetic Gradients and Decoupled Neural Interfaces. *ArXiv e-prints*.
- Dai, Z., Yang, Z., Yang, F., Cohen, W. W., and Salakhutdinov, R. (2017). Good Semi-supervised Learning that Requires a Bad GAN. *ArXiv e-prints*.
- Daniely, A., Frostig, R., and Singer, Y. (2016). Toward deeper understanding of neural networks: The power of initialization and a dual view on expressivity. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Danihelka, I., Wayne, G., Uria, B., Kalchbrenner, N., and Graves, A. (2016). Associative long short-term memory. In *the International Conference on Machine Learning (ICML)*.
- Das, A., Kottur, S., Moura, J. M. F., Lee, S., and Batra, D. (2017). Learning Cooperative Visual Dialog Agents with Deep Reinforcement Learning. *ArXiv e-prints*.

-
- De Asis, K., Hernandez-Garcia, J. F., Zacharias Holland, G., and Sutton, R. S. (2017). Multi-step Reinforcement Learning: A Unifying Algorithm. *ArXiv e-prints*.
- Deisenroth, M. P., Neumann, G., and Peters, J. (2013). A survey on policy search for robotics. *Foundations and Trend in Robotics*, 2:1–142.
- Deng, L. (2017). Three generations of spoken dialogue systems (bots), talk at AI Frontiers Conference. <https://www.slideshare.net/AIFrontiers/li-deng-three-generations-of-spoken-dialogue-systems-bots>.
- Deng, L. and Dong, Y. (2014). *Deep Learning: Methods and Applications*. Now Publishers Inc.
- Deng, L. and Li, X. (2013). Machine learning paradigms for speech recognition: An overview. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(5):1060–1089.
- Deng, L. and Liu, Y. (2017). *Deep Learning in Natural Language Processing (edited book, scheduled August 2017)*. Springer.
- Deng, Y., Bao, F., Kong, Y., Ren, Z., and Dai, Q. (2016). Deep direct reinforcement learning for financial signal representation and trading. *IEEE Transactions on Neural Networks and Learning Systems*.
- Denil, M., Agrawal, P., Kulkarni, T. D., Erez, T., Battaglia, P., and de Freitas, N. (2017). Learning to perform physics experiments via deep reinforcement learning. In *the International Conference on Learning Representations (ICLR)*.
- Denil, M., Gómez Colmenarejo, S., Cabi, S., Saxton, D., and de Freitas, N. (2017). Programmable Agents. *ArXiv e-prints*.
- Dhingra, B., Li, L., Li, X., Gao, J., Chen, Y.-N., Ahmed, F., and Deng, L. (2017). End-to-end reinforcement learning of dialogue agents for information access. In *the Association for Computational Linguistics annual meeting (ACL)*.
- Domingos, P. (2012). A few useful things to know about machine learning. *Communications of the ACM*, 55(10):78–87.
- Dong, D., Wu, H., He, W., Yu, D., and Wang, H. (2015). Multi-task learning for multiple language translation. In *the Association for Computational Linguistics annual meeting (ACL)*.
- Doshi-Velez, F. and Kim, B. (2017). Towards A Rigorous Science of Interpretable Machine Learning. *ArXiv e-prints*.
- Dosovitskiy, A. and Koltun, V. (2017). Learning to act by predicting the future. In *the International Conference on Learning Representations (ICLR)*.
- Du, S. S., Chen, J., Li, L., Xiao, L., and Zhou, D. (2017). Stochastic variance reduction methods for policy evaluation. In *the International Conference on Machine Learning (ICML)*.
- Duan, Y., Andrychowicz, M., Stadie, B. C., Ho, J., Schneider, J., Sutskever, I., Abbeel, P., and Zaremba, W. (2017). One-Shot Imitation Learning. *ArXiv e-prints*.
- Duan, Y., Chen, X., Houthoofd, R., Schulman, J., and Abbeel, P. (2016). Benchmarking deep reinforcement learning for continuous control. In *the International Conference on Machine Learning (ICML)*.
- Duan, Y., Schulman, J., Chen, X., Bartlett, P. L., Sutskever, I., and Abbeel, P. (2017). RL²: Fast reinforcement learning via slow reinforcement learning. *Submitted to Int’l Conference on Learning Representations*.
- Dulac-Arnold, G., Evans, R., van Hasselt, H., Sunehag, P., Lillicrap, T., Hunt, J., Mann, T., Weber, T., Degris, T., and Coppin, B. (2016). Deep reinforcement learning in large discrete action spaces. In *the International Conference on Machine Learning (ICML)*.

-
- El-Tantawy, S., Abdulhai, B., and Abdelgawad, H. (2013). Multiagent reinforcement learning for integrated network of adaptive traffic signal controllers (marlin-atssc): methodology and large-scale application on downtown toronto. *IEEE Transactions on Intelligent Transportation Systems*, 14(3):1140–1150.
- Eric, M. and Manning, C. D. (2017). A Copy-Augmented Sequence-to-Sequence Architecture Gives Good Performance on Task-Oriented Dialogue. *ArXiv e-prints*.
- Ernst, D., Geurts, P., and Wehenkel, L. (2005). Tree-based batch mode reinforcement learning. *The Journal of Machine Learning Research*, 6:503–556.
- Eslami, S. M. A., Heess, N., Weber, T., Tassa, Y., Szepesvári, D., Kavukcuoglu, K., and Hinton, G. E. (2016). Attend, infer, repeat: Fast scene understanding with generative models. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Fang, X., Misra, S., Xue, G., and Yang, D. (2012). Smart grid - the new and improved power grid: A survey. *IEEE Communications Surveys Tutorials*, 14(4):944–980.
- Fatemi, M., Asri, L. E., Schulz, H., He, J., and Suleman, K. (2016). Policy networks with two-stage training for dialogue systems. In *the Annual SIGdial Meeting on Discourse and Dialogue (SIGDIAL)*.
- Fernando, C., Banarse, D., Blundell, C., Zwols, Y., Ha, D., Rusu, A. A., Pritzel, A., and Wierstra, D. (2017). PathNet: Evolution Channels Gradient Descent in Super Neural Networks. *ArXiv e-prints*.
- Finn, C., Christiano, P., Abbeel, P., and Levine, S. (2016a). A connection between GANs, inverse reinforcement learning, and energy-based models. In *NIPS 2016 Workshop on Adversarial Training*.
- Finn, C. and Levine, S. (2016). Deep visual foresight for planning robot motion. *ArXiv e-prints*.
- Finn, C., Levine, S., and Abbeel, P. (2016b). Guided cost learning: Deep inverse optimal control via policy optimization. In *the International Conference on Machine Learning (ICML)*.
- Finn, C., Yu, T., Fu, J., Abbeel, P., and Levine, S. (2017). Generalizing skills with semi-supervised reinforcement learning. In *the International Conference on Learning Representations (ICLR)*.
- Firoiu, V., Whitney, W. F., and Tenenbaum, J. B. (2017). Beating the World’s Best at Super Smash Bros. with Deep Reinforcement Learning. *ArXiv e-prints*.
- Florensa, C., Duan, Y., and Abbeel, P. (2017). Stochastic neural networks for hierarchical reinforcement learning. In *the International Conference on Learning Representations (ICLR)*.
- Foerster, J., Assael, Y. M., de Freitas, N., and Whiteson, S. (2016). Learning to communicate with deep multi-agent reinforcement learning. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Foerster, J., Farquhar, G., Afouras, T., Nardelli, N., and Whiteson, S. (2017). Counterfactual Multi-Agent Policy Gradients. *ArXiv e-prints*.
- Foerster, J., Nardelli, N., Farquhar, G., Torr, P. H. S., Kohli, P., and Whiteson, S. (2017). Stabilising experience replay for deep multi-agent reinforcement learning. In *the International Conference on Machine Learning (ICML)*.
- Fortunato, M., Gheshlaghi Azar, M., Piot, B., Menick, J., Osband, I., Graves, A., Mnih, V., Munos, R., Hassabis, D., Pietquin, O., Blundell, C., and Legg, S. (2017). Noisy Networks for Exploration. *ArXiv e-prints*.
- Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., and Lempitsky, V. (2016). Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17(59):1–35.
- García, J. and Fernández, F. (2015). A comprehensive survey on safe reinforcement learning. *The Journal of Machine Learning Research*, 16:1437–1480.

-
- Garnelo, M., Arulkumaran, K., and Shanahan, M. (2016). Towards Deep Symbolic Reinforcement Learning. *ArXiv e-prints*.
- Gavrilovska, L., Atanasovski, V., Macaluso, I., and DaSilva, L. A. (2013). Learning and reasoning in cognitive radio networks. *IEEE Communications Surveys Tutorials*, 15(4):1761–1777.
- Gehring, J., Auli, M., Grangier, D., Yarats, D., and Dauphin, Y. N. (2017). Convolutional Sequence to Sequence Learning. *ArXiv e-prints*.
- Gelly, S., Schoenauer, M., Sebag, M., Teytaud, O., Kocsis, L., Silver, D., and Szepesvári, C. (2012). The grand challenge of computer go: Monte carlo tree search and extensions. *Communications of the ACM*, 55(3):106–113.
- Ghavamzadeh, M., Mannor, S., Pineau, J., and Tamar, A. (2015). Bayesian reinforcement learning: a survey. *Foundations and Trends in Machine Learning*, 8(5-6):359–483.
- Glavic, M., Fonteneau, R., and Ernst, D. (2017). Reinforcement learning for electric power system decision and control: Past considerations and perspectives. In *The 20th World Congress of the International Federation of Automatic Control*.
- Goldberg, Y. (2017). *Neural Network Methods for Natural Language Processing*. Morgan & Claypool Publishers.
- Goldberg, Y. and Kosorok, M. R. (2012). Q-learning with censored data. *Annals of Statistics*, 40(1):529–560.
- Goodfellow, I. (2017). NIPS 2016 Tutorial: Generative Adversarial Networks. *ArXiv e-prints*.
- Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press.
- Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., , and Bengio, Y. (2014). Generative adversarial nets. In *the Annual Conference on Neural Information Processing Systems (NIPS)*, page 2672?2680.
- Graves, A., Bellemare, M. G., Menick, J., Munos, R., and Kavukcuoglu, K. (2017). Automated Curriculum Learning for Neural Networks. *ArXiv e-prints*.
- Graves, A., Wayne, G., and Danihelka, I. (2014). Neural Turing Machines. *ArXiv e-prints*.
- Graves, A., Wayne, G., Reynolds, M., Harley, T., Danihelka, I., Grabska-Barwińska, A., Colmenarejo, S. G., Grefenstette, E., Ramalho, T., Agapiou, J., nech Badia, A. P., Hermann, K. M., Zwols, Y., Ostrovski, G., Cain, A., King, H., Summerfield, C., Blunsom, P., Kavukcuoglu, K., and Hassabis, D. (2016). Hybrid computing using a neural network with dynamic external memory. *Nature*, 538:471–476.
- Gregor, K., Danihelka, I., Graves, A., Rezende, D., and Wierstra, D. (2015). Draw: A recurrent neural network for image generation. In *the International Conference on Machine Learning (ICML)*.
- Grondman, I., Busoniu, L., Lopes, G. A., and Babuška, R. (2012). A survey of actor?critic reinforcement learning: Standard and natural policy gradients. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(6):1291–1307.
- Gruslys, A., Gheshlaghi Azar, M., Bellemare, M. G., and Munos, R. (2017). The Reactor: A Sample-Efficient Actor-Critic Architecture. *ArXiv e-prints*.
- Gu, S., Holly, E., Lillicrap, T., and Levine, S. (2016a). Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. *ArXiv e-prints*.
- Gu, S., Lillicrap, T., Ghahramani, Z., Turner, R. E., and Levine, S. (2017). Q-Prop: Sample-efficient policy gradient with an off-policy critic. In *the International Conference on Learning Representations (ICLR)*.
- Gu, S., Lillicrap, T., Sutskever, I., and Levine, S. (2016b). Continuous deep Q-learning with model-based acceleration. In *the International Conference on Machine Learning (ICML)*.

-
- Gulcehre, C., Chandar, S., Cho, K., and Bengio, Y. (2016). Dynamic Neural Turing Machine with Soft and Hard Addressing Schemes. *ArXiv e-prints*.
- Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. (2017). Improved Training of Wasserstein GANs. *ArXiv e-prints*.
- Gupta, A., Devin, C., Liu, Y., Abbeel, P., and Levine, S. (2017). Learning invariant feature spaces to transfer skills with reinforcement learning. In *the International Conference on Learning Representations (ICLR)*.
- Guu, K., Pasupat, P., Liu, E. Z., and Liang, P. (2017). From language to programs: Bridging reinforcement learning and maximum marginal likelihood. In *the Association for Computational Linguistics annual meeting (ACL)*.
- Haarnoja, T., Tang, H., Abbeel, P., and Levine, S. (2017). Reinforcement learning with deep energy-based policies. In *the International Conference on Machine Learning (ICML)*.
- Han, S., Mao, H., and Dally, W. J. (2016). Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding. In *the International Conference on Learning Representations (ICLR)*.
- Harrison, B., Ehsan, U., and Riedl, M. O. (2017). Rationalization: A Neural Machine Translation Approach to Generating Natural Language Explanations. *ArXiv e-prints*.
- Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer.
- Haykin, S. (2005). Cognitive radio: brain-empowered wireless communications. *IEEE Journal on Selected Areas in Communications*, 23(2):201–220.
- Haykin, S. (2008). *Neural Networks and Learning Machines (third edition)*. Prentice Hall.
- He, D., Xia, Y., Qin, T., Wang, L., Yu, N., Liu, T.-Y., and Ma, W.-Y. (2016a). Dual learning for machine translation. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- He, F. S., Liu, Y., Schwing, A. G., and Peng, J. (2017a). Learning to play in a day: Faster deep reinforcement learning by optimality tightening. In *the International Conference on Learning Representations (ICLR)*.
- He, J., Chen, J., He, X., Gao, J., Li, L., Deng, L., and Ostendorf, M. (2016b). Deep reinforcement learning with a natural language action space. In *the Association for Computational Linguistics annual meeting (ACL)*.
- He, J., Ostendorf, M., He, X., Chen, J., Gao, J., Li, L., and Deng, L. (2016c). Deep reinforcement learning with a combinatorial action space for predicting popular reddit threads. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- He, L., Lee, K., Lewis, M., and Zettlemoyer, L. (2017b). Deep semantic role labeling: What works and what’s next. In *the Association for Computational Linguistics annual meeting (ACL)*.
- He, X. and Deng, L. (2013). Speech-centric information processing: An optimization-oriented approach. *Proceedings of the IEEE — Vol. 101, No. 5, May 2013*, 101(5):1116–1135.
- Heaton, J. B., Polson, N. G., and Witte, J. H. (2016). Deep learning for finance: deep portfolios. *Applied Stochastic Models in Business and Industry*.
- Heess, N., TB, D., Sriram, S., Lemmon, J., Merel, J., Wayne, G., Tassa, Y., Erez, T., Wang, Z., Eslami, A., Riedmiller, M., and Silver, D. (2017). Emergence of Locomotion Behaviours in Rich Environments. *ArXiv e-prints*.
- Heinrich, J. and Silver, D. (2016). Deep reinforcement learning from self-play in imperfect-information games. In *NIPS 2016 Deep Reinforcement Learning Workshop*.

-
- Held, D., Geng, X., Florensa, C., and Abbeel, P. (2017). Automatic Goal Generation for Reinforcement Learning Agents. *ArXiv e-prints*.
- Henaff, M., Whitney, W. F., and LeCun, Y. (2017). Model-Based Planning in Discrete Action Spaces. *ArXiv e-prints*.
- Hester, T. and Stone, P. (2017). Intrinsically motivated model learning for developing curious robots. *Artificial Intelligence*, 247:170–86.
- Hester, T., Vecerik, M., Pietquin, O., Lanctot, M., Schaul, T., Piot, B., Sendonaris, A., Dulac-Arnold, G., Osband, I., Agapiou, J., Leibo, J. Z., and Gruslys, A. (2017). Learning from Demonstrations for Real World Reinforcement Learning. *ArXiv e-prints*.
- Hinton, G., Deng, L., Yu, D., Dahl, G. E., rahman Mohamed, A., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T. N., , and Kingsbury, B. (2012). Deep neural networks for acoustic modeling in speech recognition. *IEEE Signal Processing Magazine*, 82.
- Hirschberg, J. and Manning, C. D. (2015). Advances in natural language processing. *Science*, 349(6245):261–266.
- Ho, J. and Ermon, S. (2016). Generative adversarial imitation learning. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Ho, J., Gupta, J. K., and Ermon, S. (2016). Model-free imitation learning with policy optimization. In *the International Conference on Machine Learning (ICML)*.
- Houthoofd, R., Chen, X., Duan, Y., Schulman, J., Turck, F. D., and Abbeel, P. (2016). Vime: Variational information maximizing exploration. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Hu, Z., Yang, Z., Salakhutdinov, R., and Xing, E. P. (2017). On Unifying Deep Generative Models. *ArXiv e-prints*.
- Huang, S., Papernot, N., Goodfellow, I., Duan, Y., and Abbeel, P. (2017). Adversarial Attacks on Neural Network Policies. *ArXiv e-prints*.
- Huk Park, D., Hendricks, L. A., Akata, Z., Schiele, B., Darrell, T., and Rohrbach, M. (2016). Attentive Explanations: Justifying Decisions and Pointing to the Evidence. *ArXiv e-prints*.
- Hull, J. C. (2014). *Options, Futures and Other Derivatives (9th edition)*. Prentice Hall.
- Jaderberg, M., Mnih, V., Czarnecki, W., Schaul, T., Leibo, J. Z., Silver, D., and Kavukcuoglu, K. (2017). Reinforcement learning with unsupervised auxiliary tasks. In *the International Conference on Learning Representations (ICLR)*.
- Jaderberg, M., Simonyan, K., Zisserman, A., and Kavukcuoglu, K. (2015). Spatial transformer networks. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). *An Introduction to Statistical Learning with Applications in R*. Springer.
- Jaques, N., Gu, S., Turner, R. E., and Eck, D. (2017). Tuning recurrent neural networks with reinforcement learning. *Submitted to Int’l Conference on Learning Representations*.
- Johnson, M., Schuster, M., Le, Q. V., Krikun, M., Wu, Y., Chen, Z., Thorat, N., Viégas, F., Wattenberg, M., Corrado, G., Hughes, M., and Dean, J. (2016). Google’s Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation. *ArXiv e-prints*.
- Jordan, M. I. and Mitchell, T. (2015). Machine learning: Trends, perspectives, and prospects. *Science*, 349(6245):255–260.
- Joulin, A., Grave, E., Bojanowski, P., and Mikolov, T. (2017). Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*.

-
- Justesen, N. and Risi, S. (2017). Learning macromanagement in starcraft from replays using deep learning. In *IEEE Conference on Computational Intelligence and Games (CIG)*.
- Kadlec, R., Schmid, M., Bajgar, O., and Kleindienst, J. (2016). Text Understanding with the Attention Sum Reader Network. *ArXiv e-prints*.
- Kaelbling, L. P., Littman, M. L., and Moore, A. (1996). Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4:237–285.
- Kaiser, L. and Bengio, S. (2016). Can active memory replace attention? In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Kaiser, L., Gomez, A. N., Shazeer, N., Vaswani, A., Parmar, N., Jones, L., and Uszkoreit, J. (2017a). One Model To Learn Them All. *ArXiv e-prints*.
- Kaiser, L., Nachum, O., Roy, A., and Bengio, S. (2017b). Learning to Remember Rare Events. In *the International Conference on Learning Representations (ICLR)*.
- Kalchbrenner, N. and Blunsom, P. (2013). Recurrent continuous translation models. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- Kandasamy, K., Bachrach, Y., Tomioka, R., Tarlow, D., and Carter, D. (2017). Batch policy gradient methods for improving neural conversation models. In *the International Conference on Learning Representations (ICLR)*.
- Kansky, K., Silver, T., Mély, D. A., Eldawy, M., Lázaro-Gredilla, M., Lou, X., Dorfman, N., Sidor, S., Phoenix, S., and George, D. (2017). Schema networks: Zero-shot transfer with a generative causal model of intuitive physics. In *the International Conference on Machine Learning (ICML)*.
- Karpathy, A., Johnson, J., and Fei-Fei, L. (2016). Visualizing and understanding recurrent networks. In *ICLR 2016 Workshop*.
- Kavosh and Littman, M. L. (2017). A new softmax operator for reinforcement learning. In *the International Conference on Machine Learning (ICML)*.
- Kempka, M., Wydmuch, M., Runc, G., Toczek, J., and Jaskowski, W. (2016). ViZDoom: A Doom-based AI research platform for visual reinforcement learning. In *IEEE Conference on Computational Intelligence and Games*.
- Khandani, A. E., Kim, A. J., and Lo, A. W. (2010). Consumer credit-risk models via machine-learning algorithms. *Journal of Banking & Finance*, 34:2767–2787.
- Kim, B., massoud Farahmand, A., Pineau, J., and Precup, D. (2014). Learning from limited demonstrations. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Kingma, D. P., Rezende, D. J., Mohamed, S., and Welling, M. (2014). Semi-supervised learning with deep generative models. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., Hassabis, D., Clopath, C., Kumaran, D., and Hadsell, R. (2017). Overcoming catastrophic forgetting in neural networks. *PNAS*, 114(13):3521–3526.
- Klambauer, G., Unterthiner, T., Mayr, A., and Hochreiter, S. (2017). Self-Normalizing Neural Networks. *ArXiv e-prints*.
- Klein, G., Kim, Y., Deng, Y., Senellart, J., and Rush, A. M. (2017). OpenNMT: Open-Source Toolkit for Neural Machine Translation. *ArXiv e-prints*.
- Kober, J., Bagnell, J. A., and Peters, J. (2013). Reinforcement learning in robotics: A survey. *International Journal of Robotics Research*, 32(11):1238–1278.
- Koch, G., Zemel, R., and Salakhutdinov, R. (2015). Siamese neural networks for one-shot image recognition. In *the International Conference on Machine Learning (ICML)*.

-
- Kosorok, M. R. and Moodie, E. E. M. (2015). *Adaptive Treatment Strategies in Practice: Planning Trials and Analyzing Data for Personalized Medicine*. ASA-SIAM Series on Statistics and Applied Probability.
- Krakovsky, M. (2016). Reinforcement renaissance. *Communications of the ACM*, 59(8):12–14.
- Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Kuhn, M. and Johnson, K. (2013). *Applied Predictive Modeling*. Springer.
- Kulkarni, T. D., Narasimhan, K. R., Saeedi, A., and Tenenbaum, J. B. (2016). Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Lake, B. M., Salakhutdinov, R., and Tenenbaum, J. B. (2015). Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338.
- Lake, B. M., Ullman, T. D., Tenenbaum, J. B., and Gershman, S. J. (2016). Building Machines That Learn and Think Like People. *ArXiv e-prints*.
- Lamb, A., Goyal, A., Zhang, Y., Zhang, S., Courville, A., and Bengio, Y. (2016). Professor forcing: A new algorithm for training recurrent networks. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Lample, G. and Chaplot, D. S. (2016). Playing FPS Games with Deep Reinforcement Learning. *ArXiv e-prints*.
- Le, Q. V., Ranzato, M., Monga, R., Devin, M., Chen, K., Corrado, G. S., Dean, J., and Ng, A. Y. (2012). Building high-level features using large scale unsupervised learning. In *the International Conference on Machine Learning (ICML)*.
- LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature*, 521:436–444.
- Lee, A. X., Levine, S., and Abbeel, P. (2017). Learning visual servoing with deep features and trust region fitted Q-iteration. In *the International Conference on Learning Representations (ICLR)*.
- Lei, T., Barzilay, R., and Jaakkola, T. (2016). Rationalizing neural predictions. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- Leibo, J. Z., Zambaldi, V., Lanctot, M., Marecki, J., and Graepel, T. (2017). Multi-agent reinforcement learning in sequential social dilemmas. In *the International Conference on Autonomous Agents & Multiagent Systems (AAMAS)*.
- Levine, S., Finn, C., Darrell, T., and Abbeel, P. (2016a). End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 17:1–40.
- Levine, S., Pastor, P., Krizhevsky, A., and Quillen, D. (2016b). Learning Hand-Eye Coordination for Robotic Grasping with Deep Learning and Large-Scale Data Collection. *ArXiv e-prints*.
- Lewis, M., Yarats, D., Dauphin, Y. N., Parikh, D., and Batra, D. (2017). Deal or no deal? end-to-end learning for negotiation dialogues. In *FAIR*.
- Leyton-Brown, K. and Shoham, Y. (2008). *Essentials of Game Theory: A Concise, Multidisciplinary Introduction*. Morgan & Claypool Publishers.
- Li, J., Miller, A. H., Chopra, S., Ranzato, M., and Weston, J. (2017a). Dialogue learning with human-in-the-loop. In *the International Conference on Learning Representations (ICLR)*.
- Li, J., Miller, A. H., Chopra, S., Ranzato, M., and Weston, J. (2017b). Learning through dialogue interactions by asking questions. In *the International Conference on Learning Representations (ICLR)*.

-
- Li, J., Monroe, W., and Jurafsky, D. (2016a). A Simple, Fast Diverse Decoding Algorithm for Neural Generation. *ArXiv e-prints*.
- Li, J., Monroe, W., and Jurafsky, D. (2016b). Understanding Neural Networks through Representation Erasure. *ArXiv e-prints*.
- Li, J., Monroe, W., and Jurafsky, D. (2017a). Learning to Decode for Future Success. *ArXiv e-prints*.
- Li, J., Monroe, W., Ritter, A., Galley, M., Gao, J., and Jurafsky, D. (2016c). Deep reinforcement learning for dialogue generation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- Li, K. and Malik, J. (2017). Learning to optimize. In *the International Conference on Learning Representations (ICLR)*.
- Li, L., Chu, W., Langford, J., and Schapire, R. E. (2010). A contextual-bandit approach to personalized news article recommendation. In *the International World Wide Web Conference (WWW)*.
- Li, X., Chen, Y.-N., Li, L., and Gao, J. (2017b). End-to-End Task-Completion Neural Dialogue Systems. *ArXiv e-prints*.
- Li, X., Li, L., Gao, J., He, X., Chen, J., Deng, L., and He, J. (2015). Recurrent Reinforcement Learning: A Hybrid Approach. *ArXiv e-prints*.
- Li, X., Lipton, Z. C., Dhingra, B., Li, L., Gao, J., and Chen, Y.-N. (2016d). A User Simulator for Task-Completion Dialogues. *ArXiv e-prints*.
- Li, Y., Szepesvári, C., and Schuurmans, D. (2009). Learning exercise policies for American options. In *International Conference on Artificial Intelligence and Statistics (AISTATS09)*.
- Liang, C., Berant, J., Le, Q., Forbus, K. D., and Lao, N. (2017). Neural symbolic machines: Learning semantic parsers on freebase with weak supervision. In *the Association for Computational Linguistics annual meeting (ACL)*.
- Liang, Y., Machado, M. C., Talvitie, E., and Bowling, M. (2016). State of the art control of atari games using shallow reinforcement learning. In *the International Conference on Autonomous Agents & Multiagent Systems (AAMAS)*.
- Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. (2016). Continuous control with deep reinforcement learning. In *the International Conference on Learning Representations (ICLR)*.
- Lin, L.-J. (1992). Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine learning*, 8(3):293–321.
- Lipton, Z. C. (2016). The Mythos of Model Interpretability. *ArXiv e-prints*.
- Lipton, Z. C., Gao, J., Li, L., Li, X., Ahmed, F., and Deng, L. (2016). Efficient Exploration for Dialogue Policy Learning with BBQ Networks & Replay Buffer Spiking. *ArXiv e-prints*.
- Littman, M. L. (2015). Reinforcement learning improves behaviour from evaluative feedback. *Nature*, 521:445–451.
- Liu, B. (2012). *Sentiment Analysis and Opinion Mining*. Morgan & Claypool Publishers.
- Liu, N., Li, Z., Xu, Z., Xu, J., Lin, S., Qiu, Q., Tang, J., and Wang, Y. (2017). A hierarchical framework of cloud resource allocation and power management using deep reinforcement learning. In *37th IEEE International Conference on Distributed Computing (ICDCS 2017)*.
- Liu, S., Zhu, Z., Ye, N., Guadarrama, S., and Murphy, K. (2016). Improved Image Captioning via Policy Gradient optimization of SPIDeR. *ArXiv e-prints*.
- Liu, Y., Chen, J., and Deng, L. (2017). Unsupervised Sequence Classification using Sequential Output Statistics. *ArXiv e-prints*.

-
- Lo, A. W. (2004). The Adaptive Markets Hypothesis: Market efficiency from an evolutionary perspective. *Journal of Portfolio Management*, 30:15–29.
- Long, M., Cao, Y., Wang, J., and Jordan, M. I. (2015). Learning transferable features with deep adaptation networks. In *the International Conference on Machine Learning (ICML)*.
- Long, M., Zhu, H., Wang, J., and Jordan, M. I. (2016). Unsupervised domain adaptation with residual transfer networks. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Longstaff, F. A. and Schwartz, E. S. (2001). Valuing American options by simulation: a simple least-squares approach. *The Review of Financial Studies*, 14(1):113–147.
- Loos, S., Irving, G., Szegedy, C., and Kaliszyk, C. (2017). Deep Network Guided Proof Search. *ArXiv e-prints*.
- Lopez-Paz, D. and Ranzato, M. (2017). Gradient Episodic Memory for Continuum Learning. *ArXiv e-prints*.
- Lowe, R., Wu, Y., Tamar, A., Harb, J., Abbeel, P., and Mordatch, I. (2017). Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. *ArXiv e-prints*.
- Lu, J., Xiong, C., Parikh, D., and Socher, R. (2016). Knowing When to Look: Adaptive Attention via A Visual Sentinel for Image Captioning. *ArXiv e-prints*.
- Luenberger, D. G. (1997). *Investment Science*. Oxford University Press.
- Luo, Y., Chiu, C.-C., Jaitly, N., and Sutskever, I. (2016). Learning Online Alignments with Continuous Rewards Policy Gradient. *ArXiv e-prints*.
- Machado, M. C., Bellemare, M. G., and Bowling, M. (2017). A Laplacian framework for option discovery in reinforcement learning. In *the International Conference on Machine Learning (ICML)*.
- Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. (2017). Towards Deep Learning Models Resistant to Adversarial Attacks. *ArXiv e-prints*.
- Mahler, J., Liang, J., Niyaz, S., Laskey, M., Doan, R., Liu, X., Aparicio Ojea, J., and Goldberg, K. (2017). Dex-Net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. In *Robotics: Science and Systems (RSS)*.
- Mahmood, A. R., van Hasselt, H., and Sutton, R. S. (2014). Weighted importance sampling for off-policy learning with linear function approximation. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Manning, C. D., Raghavan, P., and Schütze, H. (2008). *Introduction to Information Retrieval*. Cambridge University Press.
- Mannion, P., Duggan, J., and Howley, E. (2016). An experimental review of reinforcement learning algorithms for adaptive traffic signal control. *Autonomic Road Transport Support Systems*, edited by McCluskey, T., Kotsialos, A., Müller, J., Klügl, F., Rana, O., and Schumann R., Springer International Publishing, Cham, pages 47–66.
- Mao, H., Alizadeh, M., Menache, I., and Kandula, S. (2016). Resource management with deep reinforcement learning. In *ACM Workshop on Hot Topics in Networks (HotNets)*.
- Mao, X., Li, Q., Xie, H., Lau, R. Y. K., and Wang, Z. (2016). Least Squares Generative Adversarial Networks. *ArXiv e-prints*.
- Matiisen, T., Oliver, A., Cohen, T., and Schulman, J. (2017). Teacher-Student Curriculum Learning. *ArXiv e-prints*.
- Maurer, A., Pontil, M., and Romera-Paredes, B. (2016). The benefit of multitask representation learning. *The Journal of Machine Learning Research*, 17(81):1–32.

-
- Merel, J., Tassa, Y., TB, D., Srinivasan, S., Lemmon, J., Wang, Z., Wayne, G., and Heess, N. (2017). Learning human behaviors from motion capture by adversarial imitation. *ArXiv e-prints*.
- Mesnil, G., Dauphin, Y., Yao, K., Bengio, Y., Deng, L., He, X., Heck, L., Tur, G., Hakkani-Tür, D., Yu, D., and Zweig, G. (2015). Using recurrent neural networks for slot filling in spoken language understanding. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(3):530–539.
- Mestres, A., Rodriguez-Natal, A., Carner, J., Barlet-Ros, P., Alarcón, E., Solé, M., Muntés, V., Meyer, D., Barkai, S., Hibbett, M. J., Estrada, G., Mañuf, K., Coras, F., Ermagan, V., Latapie, H., Cassar, C., Evans, J., Maino, F., Walrand, J., and Cabellos, A. (2016). Knowledge-Defined Networking. *ArXiv e-prints*.
- Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. In *the International Conference on Learning Representations (ICLR)*.
- Miller, T. (2017). Explanation in Artificial Intelligence: Insights from the Social Sciences. *ArXiv e-prints*.
- Miotto, R., Wang, F., Wang, S., Jiang, X., and Dudley, J. T. (2017). Deep learning for healthcare: review, opportunities and challenges. *Briefings in Bioinformatics*, pages 1–11.
- Mirhoseini, A., Pham, H., Le, Q. V., Steiner, B., Larsen, R., Zhou, Y., Kumar, N., and Mohammad Norouzi, S. Bengio, J. D. (2017). Device placement optimization with reinforcement learning. In *the International Conference on Machine Learning (ICML)*.
- Mirowski, P., Pascanu, R., Viola, F., Soyer, H., Ballard, A., Banino, A., Denil, M., Goroshin, R., Sifre, L., Kavukcuoglu, K., Kumaran, D., and Hadsell, R. (2017). Learning to navigate in complex environments. In *the International Conference on Learning Representations (ICLR)*.
- Mitra, B. and Craswell, N. (2017). Neural Models for Information Retrieval. *ArXiv e-prints*.
- Mnih, V., Badia, A. P., Mirza, M., Graves, A., Harley, T., Lillicrap, T. P., Silver, D., and Kavukcuoglu, K. (2016). Asynchronous methods for deep reinforcement learning. In *the International Conference on Machine Learning (ICML)*.
- Mnih, V., Heess, N., Graves, A., and Kavukcuoglu, K. (2014). Recurrent models of visual attention. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., and Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533.
- Mo, K., Li, S., Zhang, Y., Li, J., and Yang, Q. (2016). Personalizing a Dialogue System with Transfer Learning. *ArXiv e-prints*.
- Monroe, D. (2017). Deep learning takes on translation. *Communications of the ACM*, 60(6):12–14.
- Moody, J. and Saffell, M. (2001). Learning to trade via direct reinforcement. *IEEE Transactions on Neural Networks*, 12(4):875–889.
- Moravčík, M., Schmid, M., Burch, N., Lisý, V., Morrill, D., Bard, N., Davis, T., Waugh, K., Johanson, M., and Bowling, M. (2017). Deepstack: Expert-level artificial intelligence in heads-up no-limit poker. *Science*.
- Müller, M. (2002). Computer go. *Artificial Intelligence*, 134(1-2):145–179.
- Munos, R., Stepleton, T., Harutyunyan, A., and Bellemare, M. G. (2016). Safe and efficient off-policy reinforcement learning. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective*. The MIT Press.

-
- Nachum, O., Norouzi, M., and Schuurmans, D. (2017). Improving policy gradient by exploring under-appreciated rewards. In *the International Conference on Learning Representations (ICLR)*.
- Nachum, O., Norouzi, M., Xu, K., and Schuurmans, D. (2017). Bridging the Gap Between Value and Policy Based Reinforcement Learning. *ArXiv e-prints*.
- Narasimhan, K., Kulkarni, T., and Barzilay, R. (2015). Language understanding for text-based games using deep reinforcement learning. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- Narasimhan, K., Yala, A., and Barzilay, R. (2016). Improving information extraction by acquiring external evidence with reinforcement learning. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- Nedić, A. and Bertsekas, D. P. (2003). Least squares policy evaluation algorithms with linear function approximation. *Discrete Event Dynamic Systems: Theory and Applications*, 13:79–110.
- Neuneier, R. (1997). Enhancing q-learning for optimal asset allocation. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Neyshabur, B., Tomioka, R., Salakhutdinov, R., and Srebro, N. (2017). Geometry of Optimization and Implicit Regularization in Deep Learning. *ArXiv e-prints*.
- Ng, A. and Russell, S. (2000). Algorithms for inverse reinforcement learning. In *the International Conference on Machine Learning (ICML)*.
- Nogueira, R. and Cho, K. (2016). End-to-End Goal-Driven Web Navigation. *ArXiv e-prints*.
- Nogueira, R. and Cho, K. (2017). Task-Oriented Query Reformulation with Reinforcement Learning. *ArXiv e-prints*.
- O’Donoghue, B., Munos, R., Kavukcuoglu, K., and Mnih, V. (2017). PGQ: Combining policy gradient and q-learning. In *the International Conference on Learning Representations (ICLR)*.
- O’Donovan, P., Leahy, K., Bruton, K., and O’Sullivan, D. T. J. (2015). Big data in manufacturing: a systematic mapping study. *Journal of Big Data*, 2(20).
- Oh, J., Chockalingam, V., Singh, S., and Lee, H. (2016). Control of memory, active perception, and action in minecraft. In *the International Conference on Machine Learning (ICML)*.
- Oh, J., Guo, X., Lee, H., Lewis, R., and Singh, S. (2015). Action-conditional video prediction using deep networks in atari games. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Omidshafiei, S., Pazis, J., Amato, C., How, J. P., and Vian, J. (2017). Deep decentralized multi-task multi-agent reinforcement learning under partial observability. In *the International Conference on Machine Learning (ICML)*.
- Ontañón, S., Synnaeve, G., Uriarte, A., Richoux, F., Churchill, D., and Preuss, M. (2013). A survey of real-time strategy game ai research and competition in starcraft. *IEEE Transactions on Computational Intelligence and AI in Games*, 5(4):293–311.
- Oquab, M., Bottou, L., Laptev, I., and Sivic, J. (2015). Is object localization for free? ? weakly-supervised learning with convolutional neural networks. In *the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Osband, I., Blundell, C., Pritzel, A., and Roy, B. V. (2016). Deep exploration via bootstrapped DQN. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Pan, S. J. and Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345 – 1359.
- Papernot, N., Abadi, M., Erlingsson, Ú., Goodfellow, I., and Talwar, K. (2017). Semi-supervised knowledge transfer for deep learning from private training data. In *the International Conference on Learning Representations (ICLR)*.

-
- Papernot, N., Goodfellow, I., Sheatsley, R., Feinman, R., and McDaniel, P. (2016). cleverhans v1.0.0: an adversarial machine learning library. *ArXiv e-prints*.
- Parisotto, E., Ba, J. L., and Salakhutdinov, R. (2016). Actor-mimic: Deep multitask and transfer reinforcement learning. In *the International Conference on Learning Representations (ICLR)*.
- Parisotto, E., rahman Mohamed, A., Singh, R., Li, L., Zhou, D., and Kohli, P. (2017). Neuro-symbolic program synthesis. In *the International Conference on Learning Representations (ICLR)*.
- Paulus, R., Xiong, C., and Socher, R. (2017). A Deep Reinforced Model for Abstractive Summarization. *ArXiv e-prints*.
- Pei, K., Cao, Y., Yang, J., and Jana, S. (2017). DeepXplore: Automated Whitebox Testing of Deep Learning Systems. *ArXiv e-prints*.
- Peng, B., Li, X., Li, L., Gao, J., Celikyilmaz, A., Lee, S., and Wong, K.-F. (2017a). Composite Task-Completion Dialogue System via Hierarchical Deep Reinforcement Learning. *ArXiv e-prints*.
- Peng, P., Yuan, Q., Wen, Y., Yang, Y., Tang, Z., Long, H., and Wang, J. (2017b). Multiagent Bidirectionally-Coordinated Nets for Learning to Play StarCraft Combat Games. *ArXiv e-prints*.
- Pérez-D’Arpino, C. and Shah, J. A. (2017). C-learn: Learning geometric constraints from demonstrations for multi-step manipulation in shared autonomy. In *IEEE International Conference on Robotics and Automation (ICRA)*.
- Pfau, D. and Vinyals, O. (2016). Connecting Generative Adversarial Networks and Actor-Critic Methods. *ArXiv e-prints*.
- Phua, C., Lee, V., Smith, K., and Gayler, R. (2010). A Comprehensive Survey of Data Mining-based Fraud Detection Research. *ArXiv e-prints*.
- Pinto, L., Davidson, J., Sukthankar, R., and Gupta, A. (2017). Robust adversarial reinforcement learning. In *the International Conference on Machine Learning (ICML)*.
- Popov, I., Heess, N., Lillicrap, T., Hafner, R., Barth-Maron, G., Vecerik, M., Lampe, T., Tassa, Y., Erez, T., and Riedmiller, M. (2017). Data-efficient Deep Reinforcement Learning for Dexterous Manipulation. *ArXiv e-prints*.
- Powell, W. B. (2011). *Approximate Dynamic Programming: Solving the curses of dimensionality (2nd Edition)*. John Wiley and Sons.
- Prashanth, L., Jie, C., Fu, M., Marcus, S., and Szepesáři, C. (2016). Cumulative prospect theory meets reinforcement learning: Prediction and control. In *the International Conference on Machine Learning (ICML)*.
- Preuveneers, D. and Ilie-Zudor, E. (2017). The intelligent industry of the future: A survey on emerging trends, research challenges and opportunities in industry 4.0. *Journal of Ambient Intelligence and Smart Environments*, 9(3):287–298.
- Pritzel, A., Uria, B., Srinivasan, S., Puigdomènech, A., Vinyals, O., Hassabis, D., Wierstra, D., and Blundell, C. (2017). Neural Episodic Control. *ArXiv e-prints*.
- Provost, F. and Fawcett, T. (2013). *Data Science for Business*. O’Reilly Media.
- Puterman, M. L. (2005). *Markov decision processes : discrete stochastic dynamic programming*. Wiley-Interscience.
- Radford, A., Jozefowicz, R., and Sutskever, I. (2017). Learning to Generate Reviews and Discovering Sentiment. *ArXiv e-prints*.
- Raghu, M., Poole, B., Kleinberg, J., Ganguli, S., and Sohl-Dickstein, J. (2016). Survey of Expressivity in Deep Neural Networks. *ArXiv e-prints*.

-
- Rajendran, J., Lakshminarayanan, A., Khapra, M. M., P, P., and Ravindran, B. (2017). Attend, adapt and transfer: Attentive deep architecture for adaptive transfer from multiple sources in the same domain. *the International Conference on Learning Representations (ICLR)*.
- Ranzato, M., Chopra, S., Auli, M., and Zaremba, W. (2016). Sequence level training with recurrent neural networks. In *the International Conference on Learning Representations (ICLR)*.
- Ravi, S. and Larochelle, H. (2017). Optimization as a model for few-shot learning. In *the International Conference on Learning Representations (ICLR)*.
- Reed, S. and de Freitas, N. (2016). Neural programmer-interpreters. In *the International Conference on Learning Representations (ICLR)*.
- Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). "why should i trust you?" explaining the predictions of any classifier. In *ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)*.
- Riedmiller, M. (2005). Neural fitted Q iteration - first experiences with a data efficient neural reinforcement learning method. In *European Conference on Machine Learning (ECML)*.
- Rocktäschel, T. and Riedel, S. (2017). End-to-end Differentiable Proving. *ArXiv e-prints*.
- Ruder, S. (2017). An Overview of Multi-Task Learning in Deep Neural Networks. *ArXiv e-prints*.
- Ruelens, F., Claessens, B. J., Vandael, S., Schutter, B. D., Babuška, R., and Belmans, R. (2016). Residential demand response of thermostatically controlled loads using batch reinforcement learning. *IEEE Transactions on Smart Grid*, PP(99):1–11.
- Russell, S. and Norvig, P. (2009). *Artificial Intelligence: A Modern Approach (3rd edition)*. Pearson.
- Salakhutdinov, R. (2016). Foundations of unsupervised deep learning, a talk at Deep Learning School, <https://www.bayareadschool.org>. <https://www.youtube.com/watch?v=rK6bchqeaN8>.
- Salimans, T., Ho, J., Chen, X., and Sutskever, I. (2017). Evolution Strategies as a Scalable Alternative to Reinforcement Learning. *ArXiv e-prints*.
- Sandholm, T. (2015). Solving imperfect-information games. *Science*, 347(6218):122–123.
- Santoro, A., Raposo, D., Barrett, D. G. T., Malinowski, M., Pascanu, R., Battaglia, P., and Lillicrap, T. (2017). A simple neural network module for relational reasoning. *ArXiv e-prints*.
- Saon, G., Sercu, T., Rennie, S., and Kuo, H.-K. J. (2016). The IBM 2016 English Conversational Telephone Speech Recognition System. In *Annual Meeting of the International Speech Communication Association (INTERSPEECH)*.
- Saria, S. (2014). A \$3 trillion challenge to computational scientists: Transforming healthcare delivery. *IEEE Intelligent Systems*, 29(4):82–87.
- Schaul, T., Horgan, D., Gregor, K., and Silver, D. (2015). Universal value function approximators. In *the International Conference on Machine Learning (ICML)*.
- Schaul, T., Quan, J., Antonoglou, I., and Silver, D. (2016). Prioritized experience replay. In *the International Conference on Learning Representations (ICLR)*.
- Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117.
- Schulman, J., Abbeel, P., and Chen, X. (2017). Equivalence Between Policy Gradients and Soft Q-Learning. *ArXiv e-prints*.
- Schulman, J., Levine, S., Moritz, P., Jordan, M. I., and Abbeel, P. (2015). Trust region policy optimization. In *the International Conference on Machine Learning (ICML)*.

-
- Schuermans, D. and Zinkevich, M. (2016). Deep learning games. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Segler, M., Preuß, M., and Waller, M. P. (2017). Towards "AlphaChem": Chemical synthesis planning with tree search and deep neural network policies. In *the International Conference on Learning Representations (ICLR)*.
- Serban, I. V., Lowe, R., Charlin, L., and Pineau, J. (2015). A survey of available corpora for building data-driven dialogue systems. *arXiv e-prints*, abs/1512.05742.
- Shah, P., Hakkani-Tür, D., and Heck, L. (2016). Interactive reinforcement learning for task-oriented dialogue management. In *NIPS 2016 Deep Learning for Action and Interaction Workshop*.
- Shalev-Shwartz, S., Shamir, O., and Shammah, S. (2017). Failures of Gradient-Based Deep Learning. *ArXiv e-prints*.
- Sharma, S., Lakshminarayanan, A. S., and Ravindran, B. (2017). Learning to repeat: Fine grained action repetition for deep reinforcement learning. In *the International Conference on Learning Representations (ICLR)*.
- She, L. and Chai, J. (2017). Interactive learning for acquisition of grounded verb semantics towards human-robot communication. In *the Association for Computational Linguistics annual meeting (ACL)*.
- Shen, Y., Huang, P.-S., Gao, J., and Chen, W. (2017). Reasonet: Learning to stop reading in machine comprehension. In *ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)*.
- Shoham, Y., Powers, R., and Grenager, T. (2007). If multi-agent learning is the answer, what is the question? *Artificial Intelligence*, 171:365–377.
- Shortreed, S. M., Laber, E., Lizotte, D. J., Stroup, T. S., Pineau, J., and Murphy, S. A. (2011). Informing sequential clinical decision-making through reinforcement learning: an empirical study. *Machine Learning*, 84:109–136.
- Shwartz-Ziv, R. and Tishby, N. (2017). Opening the Black Box of Deep Neural Networks via Information. *ArXiv e-prints*.
- Silver, D. (2016). Deep reinforcement learning, a tutorial at ICML 2016. http://icml.cc/2016/tutorials/deep_rl_tutorial.pdf.
- Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. (2016a). Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489.
- Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., and Riedmiller, M. (2014). Deterministic policy gradient algorithms. In *the International Conference on Machine Learning (ICML)*.
- Silver, D., Newnham, L., Barker, D., Weller, S., and McFall, J. (2013). Concurrent reinforcement learning from customer interactions. In *the International Conference on Machine Learning (ICML)*.
- Silver, D., van Hasselt, H., Hessel, M., Schaul, T., Guez, A., Harley, T., Dulac-Arnold, G., Reichert, D., Rabinowitz, N., Barreto, A., and Degris, T. (2016b). The predictron: End-to-end learning and planning. In *NIPS 2016 Deep Reinforcement Learning Workshop*.
- Smith, L. N. (2017). Best Practices for Applying Deep Learning to Novel Applications. *ArXiv e-prints*.
- Snell, J., Swersky, K., and Zemel, R. S. (2017). Prototypical Networks for Few-shot Learning. *ArXiv e-prints*.
- Socher, R., Pennington, J., Huang, E. H., Ng, A. Y., and Manning, C. D. (2011). Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

-
- Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C., Ng, A., and Potts, C. (2013). Recursive deep models for semantic compositionality over a sentiment tree-bank. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- Spring, R. and Shrivastava, A. (2017). Scalable and sustainable deep learning via randomized hashing. In *ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)*.
- Stadie, B. C., Abbeel, P., and Sutskever, I. (2017). Third person imitation learning. In *the International Conference on Learning Representations (ICLR)*.
- Strub, F., de Vries, H., Mary, J., Piot, B., Courville, A., and Pietquin, O. (2017). End-to-end optimization of goal-driven and visually grounded dialogue systems. *ArXiv e-prints*.
- Su, P.-H., Gasic, M., Mrksic, N., Rojas-Barahona, L., Ultes, S., Vandyke, D., Wen, T.-H., and Young, S. (2016a). Continuously Learning Neural Dialogue Management. *ArXiv e-prints*.
- Su, P.-H., Gasić, M., Mrksić, N., Rojas-Barahona, L., Ultes, S., Vandyke, D., Wen, T.-H., and Young, S. (2016b). On-line active reward learning for policy optimisation in spoken dialogue systems. In *the Association for Computational Linguistics annual meeting (ACL)*.
- Sukhbaatar, S., Szlam, A., and Fergus, R. (2016). Learning multiagent communication with back-propagation. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Sukhbaatar, S., Weston, J., and Fergus, R. (2015). End-to-end memory networks. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Surana, A., Sarkar, S., and Reddy, K. K. (2016). Guided deep reinforcement learning for additive manufacturing control application. In *NIPS 2016 Deep Reinforcement Learning Workshop*.
- Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Sutton, R. (2016). Reinforcement learning for artificial intelligence, course slides. <http://www.incompleteideas.net/sutton/609%20dropbox/>.
- Sutton, R. S. (1988). Learning to predict by the methods of temporal differences. *Machine Learning*, 3(1):9–44.
- Sutton, R. S. (1990). Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *the International Conference on Machine Learning (ICML)*.
- Sutton, R. S. and Barto, A. G. (1998). *Reinforcement Learning: An Introduction*. MIT Press.
- Sutton, R. S. and Barto, A. G. (2017). *Reinforcement Learning: An Introduction (2nd Edition, in preparation)*. MIT Press.
- Sutton, R. S., Maei, H. R., Precup, D., Bhatnagar, S., Silver, D., Szepesvári, C., and Wiewiora, E. (2009a). Fast gradient-descent methods for temporal-difference learning with linear function approximation. In *the International Conference on Machine Learning (ICML)*.
- Sutton, R. S., Mahmood, A. R., and White, M. (2016). An emphatic approach to the problem of off-policy temporal-difference learning. *The Journal of Machine Learning Research*, 17:1–29.
- Sutton, R. S., McAllester, D., Singh, S., and Mansour, Y. (2000). Policy gradient methods for reinforcement learning with function approximation. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Sutton, R. S., Modayil, J., Delp, M., Degris, T., Pilarski, P. M., White, A., and Precup, D. (2011). Horde: A scalable real-time architecture for learning knowledge from unsupervised sensorimotor interaction, , proc. of 10th. In *International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*.
- Sutton, R. S., Precup, D., and Singh, S. (1999). Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1-2):181–211.

-
- Sutton, R. S., Szepesvári, C., and Maei, H. R. (2009b). A convergent $O(n)$ algorithm for off-policy temporal-difference learning with linear function approximation. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Synnaeve, G., Nardelli, N., Auvolat, A., Chintala, S., Lacroix, T., Lin, Z., Richoux, F., and Usunier, N. (2016). TorchCraft: a Library for Machine Learning Research on Real-Time Strategy Games. *ArXiv e-prints*.
- Sze, V., Chen, Y.-H., Yang, T.-J., and Emer, J. (2017). Efficient Processing of Deep Neural Networks: A Tutorial and Survey. *ArXiv e-prints*.
- Szepesvári, C. (2010). *Algorithms for Reinforcement Learning*. Morgan & Claypool.
- Tamar, A., Wu, Y., Thomas, G., Levine, S., and Abbeel, P. (2016). Value iteration networks. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Tanner, B. and White, A. (2009). RL-Glue : Language-independent software for reinforcement-learning experiments. *Journal of Machine Learning Research*, 10:2133–2136.
- Taylor, M. E. and Stone, P. (2009). Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, 10:1633–1685.
- Tesauro, G. (1994). TD-Gammon, a self-teaching backgammon program, achieves master-level play. *Neural Computation*, 6(2):215–219.
- Tessler, C., Givony, S., Zahavy, T., Mankowitz, D. J., and Mannor, S. (2017). A deep hierarchical approach to lifelong learning in minecraft. In *the AAAI Conference on Artificial Intelligence (AAAI)*.
- Theocharous, G., Thomas, P. S., and Ghavamzadeh, M. (2015). Personalized ad recommendation systems for life-time value optimization with guarantees. In *the International Joint Conference on Artificial Intelligence (IJCAI)*.
- Tian, Y., Gong, Q., Shang, W., Wu, Y., and Zitnick, L. (2017). ELF: An Extensive, Lightweight and Flexible Research Platform for Real-time Strategy Games. *ArXiv e-prints*.
- Tran, D., Hoffman, M. D., Saurous, R. A., Brevdo, E., Murphy, K., and Blei, D. M. (2017). Deep probabilistic programming. In *the International Conference on Learning Representations (ICLR)*.
- Trischler, A., Ye, Z., Yuan, X., and Suleman, K. (2016). Natural language comprehension with the epireader. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- Tsitsiklis, J. N. and Van Roy, B. (1997). An analysis of temporal-difference learning with function approximation. *IEEE Transactions on Automatic Control*, 42(5):674–690.
- Tsitsiklis, J. N. and Van Roy, B. (2001). Regression methods for pricing complex American-style options. *IEEE Transactions on Neural Networks*, 12(4):694–703.
- Usunier, N., Synnaeve, G., Lin, Z., and Chintala, S. (2017). Episodic exploration for deep deterministic policies: An application to StarCraft micromanagement tasks. In *the International Conference on Learning Representations (ICLR)*.
- van der Pol, E. and Oliehoek, F. A. (2017). Coordinated deep reinforcement learners for traffic light control. In *NIPS’16 Workshop on Learning, Inference and Control of Multi-Agent Systems*.
- van Hasselt, H., Guez, A., , and Silver, D. (2016a). Deep reinforcement learning with double Q-learning. In *the AAAI Conference on Artificial Intelligence (AAAI)*.
- van Hasselt, H., Guez, A., Hessel, M., Mnih, V., and Silver, D. (2016b). Learning values across many orders of magnitude. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention Is All You Need. *ArXiv e-prints*.

-
- Vezhnevets, A. S., Mnih, V., Agapiou, J., Osindero, S., Graves, A., Vinyals, O., and Kavukcuoglu, K. (2016). Strategic attentive writer for learning macro-actions. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Vezhnevets, A. S., Osindero, S., Schaul, T., Heess, N., Jaderberg, M., Silver, D., and Kavukcuoglu, K. (2017). Feudal networks for hierarchical reinforcement learning. In *the International Conference on Machine Learning (ICML)*.
- Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K., and Wierstra, D. (2016). Matching networks for one shot learning. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Vinyals, O., Fortunato, M., and Jaitly, N. (2015). Pointer networks. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Wang, H. and Raj, B. (2017). On the Origin of Deep Learning. *ArXiv e-prints*.
- Wang, J., Wang, W., Wang, R., and Gao, W. (2017). Beyond Monte Carlo Tree Search: Playing Go with Deep Alternative Neural Network and Long-Term Evaluation. *ArXiv e-prints*.
- Wang, J. X., Kurth-Nelson, Z., Tirumala, D., Soyer, H., Leibo, J. Z., Munos, R., Blundell, C., Kumaran, D., and Botvinick, M. (2016a). Learning to reinforcement learn. *arXiv:1611.05763v1*.
- Wang, S. I., Liang, P., and Manning, C. D. (2016b). Learning language games through interaction. In *the Association for Computational Linguistics annual meeting (ACL)*.
- Wang, W., Yang, N., Wei, F., Chang, B., and Zhou, M. (2017a). Gated self-matching networks for reading comprehension and question answering. In *the Association for Computational Linguistics annual meeting (ACL)*.
- Wang, Z., Bapst, V., Heess, N., Mnih, V., Munos, R., Kavukcuoglu, K., and de Freitas, N. (2017b). Sample efficient actor-critic with experience replay. In *the International Conference on Learning Representations (ICLR)*.
- Wang, Z., Merel, J., Reed, S., Wayne, G., de Freitas, N., and Heess, N. (2017). Robust Imitation of Diverse Behaviors. *ArXiv e-prints*.
- Wang, Z., Schaul, T., Hessel, M., van Hasselt, H., Lanctot, M., and de Freitas, N. (2016c). Dueling network architectures for deep reinforcement learning. In *the International Conference on Machine Learning (ICML)*.
- Watkins, C. J. C. H. and Dayan, P. (1992). Q-learning. *Machine Learning*, 8:279–292.
- Watter, M., Springenberg, J. T., Boedecker, J., and Riedmiller, M. (2015). Embed to control: A locally linear latent dynamics model for control from raw images. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Watters, N., Tacchetti, A., Weber, T., Pascanu, R., Battaglia, P., and Zoran, D. (2017). Visual Interaction Networks. *ArXiv e-prints*.
- Weiss, K., Khoshgoftaar, T. M., and Wang, D. (2016). A survey of transfer learning. *Journal of Big Data*, 3(9).
- Weiss, R. J., Chorowski, J., Jaitly, N., Wu, Y., and Chen, Z. (2017). Sequence-to-Sequence Models Can Directly Transcribe Foreign Speech. *ArXiv e-prints*.
- Wen, T.-H., Gasic, M., Mrksic, N., Su, P.-H., Vandyke, D., and Young, S. (2015a). Semantically conditioned LSTM-based natural language generation for spoken dialogue systems. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.
- Wen, T.-H., Vandyke, D., Mrksic, N., Gasic, M., Rojas-Barahona, L. M., Su, P.-H., Ultes, S., and Young, S. (2017). A network-based end-to-end trainable task-oriented dialogue system. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*.

-
- Wen, Z., O'Neill, D., and Maei, H. (2015b). Optimal demand response using device-based reinforcement learning. *IEEE Transactions on Smart Grid*, 6(5):2312–2324.
- Weston, J., Chopra, S., and Bordes, A. (2015). Memory networks. In *the International Conference on Learning Representations (ICLR)*.
- Whye Teh, Y., Bapst, V., Czarnecki, W. M., Quan, J., Kirkpatrick, J., Hadsell, R., Heess, N., and Pascanu, R. (2017). Distral: Robust Multitask Reinforcement Learning. *ArXiv e-prints*.
- Wiering, M. and van Otterlo, M. (2012). *Reinforcement Learning: State-of-the-Art (edited book)*. Springer.
- Williams, J. D., Asadi, K., and Zweig, G. (2017). Hybrid code networks: practical and efficient end-to-end dialog control with supervised and reinforcement learning. In *the Association for Computational Linguistics annual meeting (ACL)*.
- Williams, J. D. and Zweig, G. (2016). End-to-end LSTM-based dialog control optimized with supervised and reinforcement learning. *ArXiv e-prints*.
- Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3):229–256.
- Wilson, A. C., Roelofs, R., Stern, M., Srebro, N., and Recht, B. (2017). The Marginal Value of Adaptive Gradient Methods in Machine Learning. *ArXiv e-prints*.
- Wu, L., Xia, Y., Zhao, L., Tian, F., Qin, T., Lai, J., and Liu, T.-Y. (2017). Adversarial Neural Machine Translation. *ArXiv e-prints*.
- Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., Klingner, J., Shah, A., Johnson, M., Liu, X., Kaiser, L., Gouws, S., Kato, Y., Kudo, T., Kazawa, H., Stevens, K., Kurian, G., Patil, N., Wang, W., Young, C., Smith, J., Riesa, J., Rudnick, A., Vinyals, O., Corrado, G., Hughes, M., and Dean, J. (2016). Google's neural machine translation system: Bridging the gap between human and machine translation. *ArXiv e-prints*.
- Wu, Y. and Tian, Y. (2017). Training agent for first-person shooter game with actor-critic curriculum learning. In *the International Conference on Learning Representations (ICLR)*.
- Xiong, C., Zhong, V., and Socher, R. (2017a). Dynamic coattention networks for question answering. In *the International Conference on Learning Representations (ICLR)*.
- Xiong, W., Droppo, J., Huang, X., Seide, F., Seltzer, M., Stolcke, A., Yu, D., and Zweig, G. (2017b). The microsoft 2016 conversational speech recognition system. In *The IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.
- Xu, K., Ba, J. L., Kiros, R., Cho, K., Courville, A., Salakhutdinov, R., Zemel, R. S., and Bengio, Y. (2015). Show, attend and tell: Neural image caption generation with visual attention. In *the International Conference on Machine Learning (ICML)*.
- Xu, L. D., He, W., and Li, S. (2014). Internet of things in industries: A survey. *IEEE Transactions on Industrial Informatics*, 10(4):2233–2243.
- Yahya, A., Li, A., Kalakrishnan, M., Chebotar, Y., and Levine, S. (2016). Collective robot reinforcement learning with distributed asynchronous guided policy search. *ArXiv e-prints*.
- Yang, X., Chen, Y.-N., Hakkani-Tur, D., Crook, P., Li, X., Gao, J., and Deng, L. (2016). End-to-End Joint Learning of Natural Language Understanding and Dialogue Manager. *ArXiv e-prints*.
- Yang, Z., He, X., Gao, J., Deng, L., and Smola, A. (2015). Stacked Attention Networks for Image Question Answering. *ArXiv e-prints*.
- Yang, Z., Hu, J., Salakhutdinov, R., and Cohen, W. W. (2017). Semi-supervised qa with generative domain-adaptive nets. In *the Association for Computational Linguistics annual meeting (ACL)*.
- Yannakakis, G. N. and Togelius, J. (2017). *Artificial Intelligence and Games*. (draft).

-
- Yao, H., Szepesvari, C., Sutton, R. S., Modayil, J., and Bhatnagar, S. (2014). Universal option models. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Yogatama, D., Blunsom, P., Dyer, C., Grefenstette, E., and Ling, W. (2017). Learning to compose words into sentences with reinforcement learning. In *the International Conference on Learning Representations (ICLR)*.
- Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. (2014). How transferable are features in deep neural networks? In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Young, S., Gašić, M., Thomson, B., and Williams, J. D. (2013). POMDP-based statistical spoken dialogue systems: a review. *PROC IEEE*, 101(5):1160–1179.
- Yu, L., Zhang, W., Wang, J., and Yu, Y. (2017). Seqgan: Sequence generative adversarial nets with policy gradient. In *the AAAI Conference on Artificial Intelligence (AAAI)*.
- Yu, Y.-L., Li, Y., Szepesvári, C., and Schuurmans, D. (2009). A general projection property for distribution families. In *the Annual Conference on Neural Information Processing Systems (NIPS)*.
- Zagoruyko, S. and Komodakis, N. (2017). Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In *the International Conference on Learning Representations (ICLR)*.
- Zaremba, W. and Sutskever, I. (2015). Reinforcement Learning Neural Turing Machines - Revised. *ArXiv e-prints*.
- Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. (2017). Understanding deep learning requires rethinking generalization. In *the International Conference on Learning Representations (ICLR)*.
- Zhang, H., Yu, H., and Xu, W. (2017a). Listen, Interact and Talk: Learning to Speak via Interaction. *ArXiv e-prints*.
- Zhang, J., Ding, Y., Shen, S., Cheng, Y., Sun, M., Luan, H., and Liu, Y. (2017b). THUMT: An Open Source Toolkit for Neural Machine Translation. *ArXiv e-prints*.
- Zhang, Y., Mustafizur Rahman, M., Braylan, A., Dang, B., Chang, H.-L., Kim, H., McNamara, Q., Angert, A., Banner, E., Khetan, V., McDonnell, T., Thanh Nguyen, A., Xu, D., Wallace, B. C., and Lease, M. (2016). Neural Information Retrieval: A Literature Review. *ArXiv e-prints*.
- Zhang, Y., Pezeshki, M., Brakel, P., Zhang, S., Yoshua Bengio, C. L., and Courville, A. (2017c). Towards End-to-End Speech Recognition with Deep Convolutional Neural Networks. *ArXiv e-prints*.
- Zhao, T. and Eskenazi, M. (2016). Towards end-to-end learning for dialog state tracking and management using deep reinforcement learning. In *the Annual SIGdial Meeting on Discourse and Dialogue (SIGDIAL)*.
- Zhou, H., Huang, M., Zhang, T., Zhu, X., and Liu, B. (2017). Emotional Chatting Machine: Emotional Conversation Generation with Internal and External Memory. *ArXiv e-prints*.
- Zhou, Z.-H. (2016). *Machine Learning (in Chinese)*. Tsinghua University Press, Beijing, China.
- Zhou, Z.-H. and Feng, J. (2017). Deep Forest: Towards An Alternative to Deep Neural Networks. *ArXiv e-prints*.
- Zhu, X. and Goldberg, A. B. (2009). *Introduction to semi-supervised learning*. Morgan & Claypool.
- Zhu, Y., Mottaghi, R., Kolve, E., Lim, J. J., Gupta, A., Li, F.-F., and Farhadi, A. (2016). Target-driven Visual Navigation in Indoor Scenes using Deep Reinforcement Learning. *ArXiv e-prints*.
- Zinkevich, M. (2017). *Rules of Machine Learning: Best Practices for ML Engineering*. http://martin.zinkevich.org/rules_of_ml/rules_of_ml.pdf.
- Zoph, B. and Le, Q. V. (2017). Neural architecture search with reinforcement learning. In *the International Conference on Learning Representations (ICLR)*.