DEIM2021 E31-4

不完全エージェントが混在した軌跡における模倣学習の評価

木村　大地†　　泉谷　知範†　　島田健一郎†　　加嶋　健司††

† NTT コミュニケーションズ株式会社　〒 108–6311 東京都港区芝浦 3-4-1
†† 京都大学大学院 情報学研究科　〒 606–8501 京都府京都市左京区吉田本町
E-mail: †{d.kimura,tomonori.izumitani,kenichiro.shimada}@ntt.com, ††kashima@amp.i.kyoto-u.ac.jp

あらまし　模倣学習は強化学習の一分野として, 報酬設計を必要とせず手本となるエキスパートの状態-行動の履歴（軌跡）のみから方策を学習する手法である. このため, 強化学習のサンプル効率の悪さを緩和し, 実世界適用の手法として注目されている. 現実の問題設定では, 完全なエキスパートのみから軌跡が生成されることは少なく, 方策の異なるエキスパートや不完全なエージェントの軌跡が混在している場合が多々ある. このため, 軌跡に応じて模倣学習される方策は一種類の軌跡をもちいた場合とは異なる特徴をもつことが予想される. 本研究では, 性質の異なる複数の方策から生成された軌跡を用いて模倣学習された方策の評価と考察を行う.

キーワード　模倣学習, 強化学習, 制御, 機械学習

1 背　　景

近年, 強化学習は深層学習手法の発展に伴い, 人間をはるかに上回るような大きな成果をあげている [1] [2] [3]. 強化学習は, 行動主体であるエージェントと制御対象の環境により定式化される. エージェントは方策 $\pi(a|s)$ で表される行動基準に基づき, 環境へ行動を送出し, その結果として次の状態 s' と行動の評価基準である報酬 r' を受け取る. エージェントは将来にわたって得られる報酬を最大化するように方策を更新する. このようなエージェントと環境の相互作用のなかでデータを反復的に収集しながら, 最適な制御則（方策）を探索する. 先行研究 [1] [2] [3] が示すように, 強化学習は環境のダイナミクスを陽として扱わない（行動 a の実績値 s', r' のみを使用する）ため, 環境のダイナミクスが高次元であったり, 未知または不完全にしかわからない場合においても, 良好な結果を示している.

しかし, 強化学習は環境との相互作用の中でデータを収集しながら学習を行うオンラインな手法であるため, 現実世界での活用を考えると経済・時間的コスト・安全性の関係から難しい場合がある [4] . 例えば, 自動運転, プラント制御, 金融の分野では, 学習初期のエージェントのランダムな行動は大きなリスクを伴い危険である. また, 試行錯誤が可能な環境であっても, サンプル効率（学習効率）改善の観点から事前に収集済みデータから学習することが望ましい. さらに, 別の問題として行動基準を意味する報酬の設計自体が難しい場合も多々ある. 自動運転を例とすれば, 道路状況は時々刻々し, 平常運転であるのか事故につながる運転であるのかを判別する適切な報酬を設計することは, 関係するパラメータが多数存在するため困難である.

そのため, すでに収集済みデータから最適な方策を学習するオフライン強化学習（バッチ強化学習）手法が注目されている [4] . その中でも, 模倣学習は報酬関数を明示的に設定する必要なく, 手本となる状態-行動対から方策を学習することができるため, 有用であると考えられる.

既存の模倣学習手法では, 学習にもちいるデータは報酬を最大化するように最適化された方策（エキスパート）のみから生成されると仮定する場合が多い. しかし, 現実の問題設定では, エキスパートのみから軌跡が生成されることは少なく, 方策が異なったり, 不完全なエージェントの軌跡が混在している場合が多々ある. このため, 学習にもちいる軌跡に応じて, 模倣学習で復元される方策は 1 種類のエキスパートの場合とは異なる特徴をもつことが予想される. 先行研究では, 不完全な軌跡に対応した模倣学習手法 [5] が提案されているが, データセット自体に着目した研究は少ない.

そこで, 本研究では性質の異なる複数方策から生成された軌跡を用いて模倣学習を行い, 学習された方策の評価と考察を行った. 具体的には, 不十分なデータ量のもとで模倣学習した際に, どのような方策が獲得されるかを一般の問題設定で理論的に解析することは困難であるため, 3.1 節では線形システムを対象とすることで解析的な考察を行い, 3.2 節ではある非線形システムを対象として数値シミュレーションにより比較検討を行う.

2 準　　備

2.1 強 化 学 習

離散時間マルコフ決定過程（Markov Decision Process, MDP）を仮定し, $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \gamma)$ と定義する. ここで, $\mathcal{S}$ は状態 $s \in \mathcal{S}$ の組, $\mathcal{A}$ は行動 $a \in \mathcal{A}$ の組, $P(s'|s,a)$ は環境のダイナミクスを意味する状態遷移確率, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ は報酬関数, $\gamma \in (0, 1]$ は割引率である. ある時刻 t における状態, 行動を, それぞれ s_t, a_t とする.

強化学習の問題設定では, エージェントの方策 $\pi(a|s)$ として, 次の期待収益を最大化する最適方策 π^* を求めることになる.

$$J(\pi) = \mathbb{E}\left[\sum_{t=0}^{T} \gamma^t r(s_t, a_t)\right] \tag{1}$$

2.2 模倣学習

模倣学習は, 行動軌跡から学習する手法である. 代表的な模倣学習手法として, 敵対的モデルに基づく GAIL [6] が提案されているが, これは追加の環境との相互作用により方策と報酬関数を学習する手法である. 本研究では, 過去の行動履歴のみから方策を復元することを目的とするため Behavioral Cloning (BC) を用いる. BC では教師あり学習の要領で, 軌跡 $\mathcal{D} = \{(s_i, a_i), i = 0, \cdots, N\}$ から状態 s_t に対応する $\tilde{a}_t$ を学習する. ϕ でパラメータ化された方策 π_ϕ として, 最尤推定により最適なパラメータ ϕ^* を求める [7].

$$\phi^* = \arg\max_{\phi} \prod_{i=0}^{N} \pi_\phi(\tilde{a}_i|s_i) \tag{2}$$

3 実　　験

本研究では, BC で学習した方策の行動軌跡に対する依存性を検証した. BC に用いる軌跡の性質を変化させるため, 2 つの異なる方策からデータを生成した. それぞれについて, 行動に対する制約が厳しいが加わるノイズが小さい方策をエキスパート, 制約は小さいがノイズが大きい方策をアマチュアと呼び, 添字に ex, ama を用いる. 生成された軌跡の混合割合を変化させ, BC を行い, 学習ずみ方策の性能を検証した. この方策をスチューデントと呼び, 添字に st を用いる.

制御対象である環境は, 線形, 非線形システムを対象とした. 方策が解析的に求めることのできる線形システムで検証した後, 非線形システムへ拡張し検証を行った.

3.1 線形システム

まず, エキスパート-アマチュアの軌跡の混合比率による BC への影響を調査するため, 以下で示される 2 次元離散時間線形システムを仮定する.

$$s_{t+1} = As_t + Ba_t + w_t \tag{3}$$

$$A = \begin{bmatrix} 1 & 0.5 \\ 0.2 & 1 \end{bmatrix},\ B = \begin{bmatrix} 1 \\ 0 \end{bmatrix},\ w_t \sim \mathcal{N}(0, \sigma^2) \tag{4}$$

σ^2 は系に加わるノイズの分散パラメータである. この系において, 初期状態 $s_0 = [1, 4]^T$ とし, 原点 $[0, 0]^T$ に到達することを目的とする. このとき, 最適な方策が満たすべきコスト関数を次のように定義する.

$$J = \sum_{k=0}^{T-1} (s_t^{\mathrm{T}} Q s_t + a_t^{\mathrm{T}} R a_t) + s_T^{\mathrm{T}} Q_T s_T \tag{5}$$

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},\ Q_T = \begin{bmatrix} 50 & 0 \\ 0 & 50 \end{bmatrix},\ R = 10 \tag{6}$$

第 1 項, 第 2 項は, それぞれ各時刻における状態 s_t, 行動 a_t に応じたコスト, 第 3 項は終端時刻 $t = T$ における状態に対するコストである. このコスト関数を最小化することは, なるべく

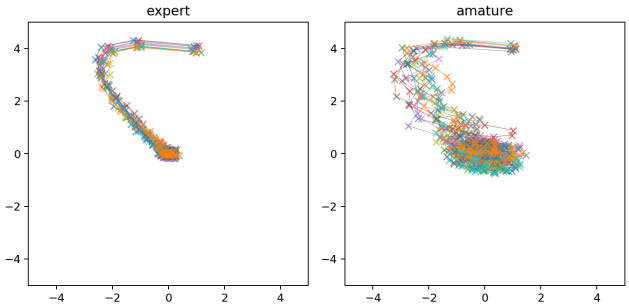

Figure 1　BC に用いたエキスパートとアマチュアの軌跡. エキスパートの軌跡はノイズの影響が小さいため, 軌跡が大きく分散することなく, 原点へと収束する.

系に対する入力量（行動）を小さくしながら原点へ到達するような方策を求めることになる.

コスト関数を最小化する方策 K は解析的に求まり, 次のように表される.

$$\begin{aligned} a_t &= -K_t s_t \\ K_t &= (R + B^{\mathrm{T}} P_t B)^{-1} B^{\mathrm{T}} P_t A \end{aligned} \tag{7}$$

ただし, 行列 P は終端条件 $P_T = Q_T$ のもとでの離散時間代数リッカチ方程式の解である.

$$P_{t-1} = A^{\mathrm{T}} P_t A - A^{\mathrm{T}} P_t B (R + B^{\mathrm{T}} P_t B)^{-1} B^{\mathrm{T}} P_t A + Q \tag{8}$$

本研究では, エキスパート, アマチュアそれぞれについて系に加わるノイズと行動に対するコストを変化させることでコスト関数を定義した. エキスパートについては, 加わるノイズが小さいが行動コストが大きく, アマチュアについては加わるノイズが大きいが行動コストが小さいと仮定し, 式 (4), (7) のそれぞれのパラメータは以下のように設定した.

$$\sigma^{\mathrm{ex}} = 0.1,\ R^{\mathrm{ex}} = R/\sigma^{\mathrm{ex}} \tag{9}$$

$$\sigma^{\mathrm{ama}} = 0.5,\ R^{\mathrm{ama}} = R/\sigma^{\mathrm{ama}} \tag{10}$$

以上の条件のもと, エキスパート, アマチュアそれぞれについて 10 回の試行から軌跡を生成した (Fig.1).

この軌跡に基づき, BC を行った. BC によって学習される制御則は, 各時刻の状態 s_t におけるエキスパート-アマチュアの尤度の比で定義する. 尤度は

$$P^{\mathrm{ex}}(s_t|\mu_t^{\mathrm{ex}}, \Sigma_t^{\mathrm{ex}}) = \mathcal{N}(s_t|\mu_t^{\mathrm{ex}}, \Sigma_t^{\mathrm{ex}}) \tag{11}$$

$$P^{\mathrm{ama}}(s_t|\mu_t^{\mathrm{ama}}, \Sigma_t^{\mathrm{ama}}) = \mathcal{N}(s_t|\mu_t^{\mathrm{ama}}, \Sigma_t^{\mathrm{ama}}) \tag{12}$$

であり, μ, Σ はそれぞれ, 軌跡の平均値と分散である. BC にもちいるエキスパート-アマチュアの軌跡の混合比率を η として, 制御則 K^{st} は次式となる.

$$a_t^{\mathrm{st}} = -K_t^{\mathrm{st}} s_t \tag{13}$$

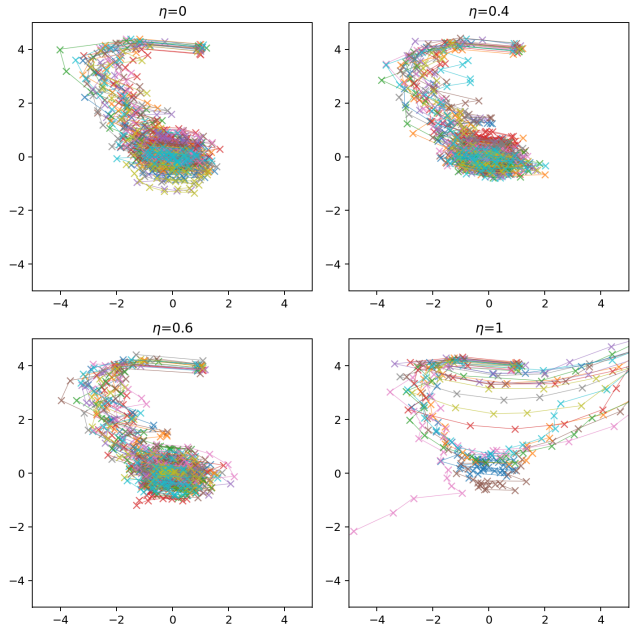

Figure 2　データの混合比率 η を変化させ, 制御則を学習した結果. エキスパートのみで学習を行うとノイズの影響を受けて発散する.

$$K_t^{\mathrm{st}} = \frac{\eta P_t^{\mathrm{ex}}}{\eta P_t^{\mathrm{ex}} + (1-\eta) P_t^{\mathrm{ama}}} K_t^{\mathrm{ex}} + \frac{(1-\eta) P_t^{\mathrm{ama}}}{\eta P_t^{\mathrm{ex}} + (1-\eta) P_t^{\mathrm{ama}}} K_t^{\mathrm{ama}} \tag{14}$$

ここで, $\eta \to 1$ のときエキスパートの軌跡のみで学習した制御則, $\eta \to 0$ のときアマチュアのみの軌跡で学習した制御則になる.

Fig.2 に η を変えて BC を行った結果を示す. アマチュアのデータが含まれている場合には, ノイズの影響を受けながらも原点へと収束することが確認できる. 一方, エキスパートの軌跡のみから BC を行った場合には, エキスパートの訪問していない状態に達すると元の軌跡に戻るための行動がわからず, 発散してしまうことがわかる. これは, 模倣学習に用いたデータセットの状態-行動分布と, 学習後の方策が直面する状態の分布が異なるために性能が劣化する Distribution shift [8] [9] のためであると考えられる. 以上より, 線形なシステムにおいて解析的に BC を行った場合においても, 学習データに依存して性能が変化することが示された.

3.2　非線形システム

環境として OpenAI Gym の LunarLanderContinuous-v2 [10] を用いた（以後, LunarLander と略記する）. 方策や BC の実装には Python3.8, stable-baselines3 [11], Pytorch [12] を用いた.

LunarLander の模式図を Fig.3 に示す. 制御対象の Space ship は初期位置 $(0, 1.5)$ から出発し, 左右下方向にジェットを噴射しながら, 目標着陸地点 $(-0.1, 0), (0.1, 0)$ 間に着陸することを目的とする. エージェントの取りうる行動 a は下方向と左右方向へのジェット噴射量（連続値）で $a = \{[-1, 1], [-1, 1]\}$ の 2 次元,

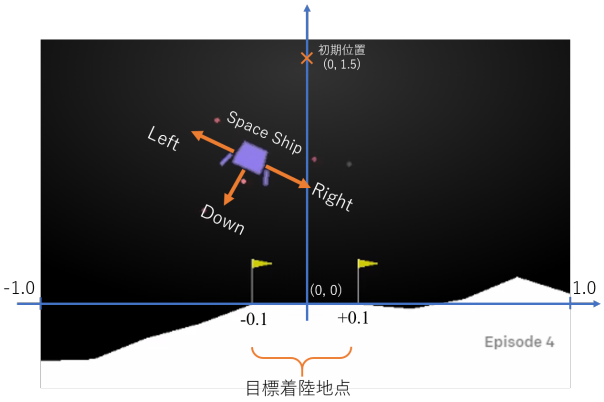

Figure 3　LunarLanderContinuous-v2 の模式図.

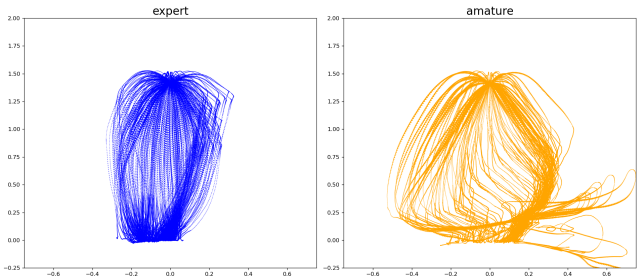

Figure 4　π^{ex}, π^{ama} で生成した軌跡. それぞれの点は速度ベクトルを示す.

状態 s は Space ship の座標 (x, y), 速度 $(\dot{x}, \dot{y})$, 回転角 θ, 角速度 $\dot{\theta}$ として, $s = \{x, y, \dot{x}, \dot{y}, \theta, \omega, (\text{if 右足接地 : 1 else : 0}), (\text{if 左足接地 : 1 else : 0})\}$ の 8 次元で表される. また, 報酬関数は次式で定義される.

$$\begin{aligned} r_t = {} & \mathrm{shape}_t - \mathrm{shape}_{t-1} \\ & - 0.3 \times (\text{下方向へのジェット}) \\ & - 0.03 \times (\text{左右方向へのジェット}) \end{aligned} \tag{15}$$

$$\begin{aligned} \mathrm{shape}_t = {} & -100\sqrt{x^2 + y^2} - 100\sqrt{\dot{x}^2 + \dot{y}^2} - 100|\theta| \\ & + 10 \times (\text{if 右足接地 : 1 else : 0}) \\ & + 10 \times (\text{if 左足接地 : 1 else : 0}) \end{aligned}$$

各時間ステップで姿勢が改善するごとに小さい報酬が得られ, さらに目標着陸地点間に着陸成功時に+100, 失敗時に-100 の報酬が得られる. また, ジェットを噴射するごとにペナルティを受ける. つまり, なるべく姿勢を安定させ, 燃料使用を少なくしながら, 目標着陸地点間に着陸させることが目的となる.

まず, この環境の下で BC のためのデータセットとなる軌跡 $\mathcal{D}$ を強化学習により生成した. LunarLander の実装のまま強化学習した方策を π^{ex}, ジェット噴射に対するペナルティをなくした報酬関数, すなわち式 (15) において右辺第 3, 4 項を 0 にし, 行動にノイズを加えた環境のもとで学習した方策を π^{ama} とする. $\mathcal{D}$ の生成にもちいる方策の学習には, 代表的な深層強化学習手法である Soft Actor-Critic [13] を用いた. 同じハイ

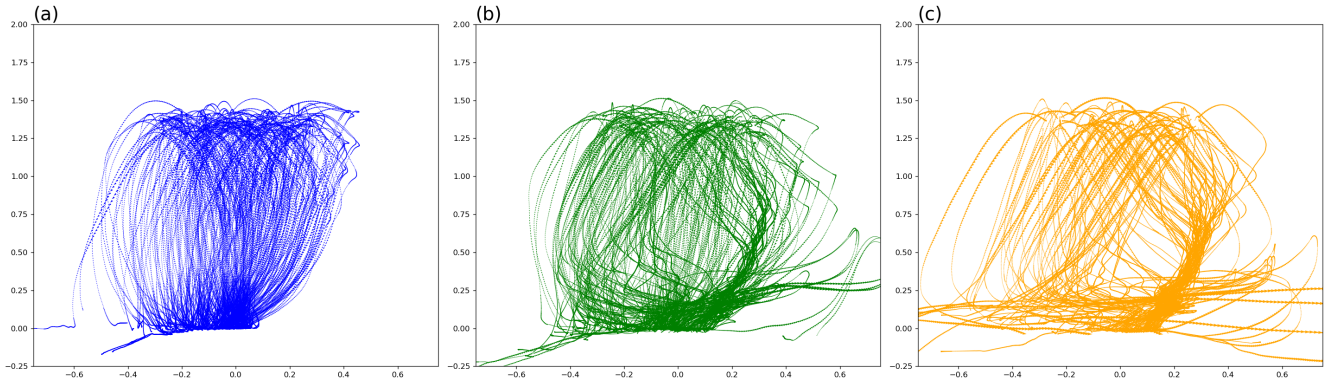

Figure 5　BC で学習した方策の検証結果. (a) エキスパートの軌跡のみから BC した π^{st}, (b) エキスパートとアマチュアの混合軌跡から BC した π^{st}, (c) アマチュアの軌跡のみから BC した π^{st} の描く軌跡.

Table 1　それぞれの方策の収益の平均値

expert	mixed	amature
262.9 ± 41.7	230.2 ± 75.1	142.5 ± 119.6

パーパラメータの下で十分回（5×10^6 ステップ）の学習を行い, 最も平均収益の高い方策を π^{ex}, π^{ama} とした. これらの方策を用いて, 8×10^5 個の軌跡を作成し, $\mathcal{D}^{\mathrm{ex}} = \{(s_i^{\mathrm{ex}}, a_i^{\mathrm{ex}}), i = 1, \cdots, 8 \times 10^5\}, \mathcal{D}^{\mathrm{ama}} = \{(s_i^{\mathrm{ama}}, a_i^{\mathrm{ama}}), i = 1, \cdots, 8 \times 10^5\}$ とした. それぞれの軌跡を Fig.4 に示す.

生成した軌跡 $\mathcal{D} = \mathcal{D}^{\mathrm{ex}} \cup \mathcal{D}^{\mathrm{ama}}$ から, BC により π^{st} の学習を行った. π^{st} は中間層が 2 層でそれぞれのノード数が 256, 活性化関数が ReLU のニューラルネットワークとし, 最適化には AdaDelta [14] を用いた. LunarLander 環境では連続行動値を扱うため, この方策を表現するニューラルネットワークは状態 s を入力とし, ガウス分布の平均分散パラメータを出力する.

BC により学習した π^{st} をもちいて, LunarLander 環境下で検証を行った. π^{st} の頑健性を検証するため, Space Ship の初期座標をランダムに変化させ, エピソードごとに得られる収益の平均と標準偏差を計算した. 初期位置は, 一様分布 U に従い, $(x, y) = (U(-0.2, 0.2), 1.5 - U(0, 0.15))$ とした. この条件のもとで, BC にもちいるデータを変化させ, π^{st} の軌跡と収益の比較を行った.

4　結　　果

Fig.5 に BC により学習した π^{st} を用いて, LunarLander 環境下で検証を行った結果を示す. Fig.5(a)(b)(c) はそれぞれ, エキスパートの軌跡のみ, エキスパートとアマチュアの混合軌跡, アマチュアの軌跡のみから BC で学習した π^{st} の軌跡である. また, それぞれの場合について, 収益の平均値と標準偏差を Table1 に示す.

Fig.5 の軌跡と Table 1 の結果ついて, エキスパートの軌跡のみから学習した方策が最もよい性能を示しており, アマチュアの軌跡のみのものは多くの場合で目標への着陸に失敗していることがわかる. また, 混合した軌跡のものは 2 つの中間程度の性能を示した. 特に, $(0.4, 0.5)$ 付近の軌跡は, 学習にもちいたアマチュア (Fig.4) の特性に影響を受けていることが考えられる. 一方で, Fig.5(a) において, BC 後の方策は学習に用いた軌跡の範囲外においても適切な制御が行えており, これは 3.1 で示した結果や先行研究 [8] [9] の結果とは異なる. 本研究では Fig.4, 5 に示すように, 多次元状態 s_t のうち, (x, y) 座標についてのみに着目し観察を行った. しかし, 実際に BC で学習した方策が行動を決定するのにもちいている重要な特徴量は, (x, y) 座標以外の別の特徴であることが考えられる. 今後, どのような特徴量に依存して, BC の性能が決定されるのかをさらに検討したい.

5　結　　論

本研究では, 報酬関数や外乱の大きさの異なる学習済みエージェントから生成した軌跡に基づき, Behavioral Cloning で方策 π^{st} を学習, 線形/非線形システムを対象として検証を行った. その結果, 線形システムにおいては, エキスパート軌跡のみから BC をした場合よりも, 2 つのエージェントの軌跡を混合することで, 性能改善がみられた. この結果は先行研究で示された Distribution shift と対応する. 一方で, 非線形システムの場合は, エキスパートの軌跡のみから学習した方策 π^{st} が最もよい性能を示した. つまり, 線形システム/非線形システムの場合で BC は異なるを示しており, この違いについてさらなる研究を行う予定である.

文　　献

[1] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. A. Riedmiller, “Playing atari with deep reinforcement learning,” *arXiv preprint arXiv:1312.5602*, 2013.

[2] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, “Mastering the game of go with deep neural networks and tree search,” *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[3] M. Hessel, J. Modayil, H. van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. G. Azar, and

D. Silver, "Rainbow: Combining improvements in deep reinforcement learning," in *AAAI*, pp. 3215–3222, 2017.

[4] S. Levine, A. Kumar, G. Tucker, and J. Fu, "Offline reinforcement learning: Tutorial, review, and perspectives on open problems," *arXiv preprint arXiv:2005.01643*, 2020.

[5] M. Sun and X. Ma, "Adversarial imitation learning from incomplete demonstrations," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, pp. 3513–3519, 2019.

[6] J. Ho and S. Ermon, "Generative adversarial imitation learning," in *Advances in Neural Information Processing Systems*, vol. 29, pp. 4565–4573, 2016.

[7] F. Torabi, G. Warnell, and P. Stone, "Behavioral cloning from observation," in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, pp. 4950–4957, 2018.

[8] S. Ross and D. Bagnell, "Efficient reductions for imitation learning," in *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS) 2010*, pp. 661–668, 2010.

[9] S. Ross, G. J. Gordon, and J. A. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, vol. 15, pp. 627–635, 2011.

[10] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," 2016.

[11] A. Raffin, A. Hill, M. Ernestus, A. Gleave, A. Kanervisto, and N. Dormann, "Stable baselines3." `https://github.com/DLR-RM/stable-baselines3`, 2019.

[12] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems*, vol. 32, pp. 8026–8037, 2019.

[13] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International Conference on Machine Learning*, pp. 1856–1865, 2018.

[14] M. D. Zeiler, "Adadelta: An adaptive learning rate method," *arXiv preprint arXiv:1212.5701*, 2012.