Policy Caches with Successor Features

Mark Nemecek [1] **Ronald Parr** [1]

A. Theoretical Proof

Let Λ^{π_i} be a matrix with each row corresponding to a state $s \in \mathcal{S}$, and each column corresponding to a successor feature. Thus, each row corresponds to a row vector of successor features for a state under π_i: $\Lambda^{\pi_i}(s) = \psi^{\pi_i}(s, \pi_i(s))$ and is analogous to the relationship between V^{π_i} and Q^{π_i}.

Definition 1. *An approximate successor features matrix $\tilde{\Lambda}^{\pi_i}$ differs from the exact matrix Λ^{π_i} by a state-wise error matrix $\Delta^{\pi_i} = \tilde{\Lambda}^{\pi_i} - \Lambda^{\pi_i}$.*

Definition 2. *If $\boldsymbol{w}_j$ is the weight vector for task j, then the approximate value function matrix induced by $\tilde{\Lambda}^{\pi_i}$ in task j is the column vector $\tilde{V}_j^{\pi_i} = \tilde{\Lambda}^{\pi_i}\boldsymbol{w}_j$.*

Definition 3. *The state-wise approximation error of $\tilde{V}_j^{\pi_i}$ is the column vector $\boldsymbol{\epsilon}_j^{\pi_i} = |\tilde{V}_j^{\pi_i} - V_j^{\pi_i}| = |\Delta^{\pi_i}\boldsymbol{w}_j|$ where $|\cdot|$ denotes the element-wise absolute value.*

Theorem 1. *If π_R is an optimal policy under r_R and r_R is a positive conical combination, i.e., $\alpha_i \geq 0$ and $\alpha = \sum_{i \in \mathcal{T}} \alpha_i > 0$, then*

$$\begin{aligned}\max_{i \in \mathcal{T}} \left[\tilde{V}_R^{\pi_i}(s) - \boldsymbol{\epsilon}_R^{\pi_i}(s)\right] \leq \\ V_R^{\pi_R}(s) \leq \\ \sum_{i \in \mathcal{T}} \alpha_i \tilde{V}_i^{\pi_i}(s) + \sum_{i \in \mathcal{T}} \alpha_i \boldsymbol{\epsilon}_i^{\pi_i}(s)\end{aligned}$$

Proof. From the definition of an optimal policy:

$$\begin{aligned}V_R^{\pi_i} &\leq V_R^{\pi_R} \qquad\qquad \forall\, i \in \mathcal{T}\\ \sum_{i \in \mathcal{T}} \alpha_i V_R^{\pi_i} &\leq \sum_{i \in \mathcal{T}} \alpha_i V_R^{\pi_R}\\ \sum_{i \in \mathcal{T}} \alpha_i V_R^{\pi_i} &\leq V_R^{\pi_R} \sum_{i \in \mathcal{T}} \alpha_i\\ \sum_{i \in \mathcal{T}} \alpha_i V_R^{\pi_i} &\leq \alpha V_R^{\pi_R}\\ \frac{1}{\alpha} \sum_{i \in \mathcal{T}} \alpha_i V_R^{\pi_i} &\leq V_R^{\pi_R} \qquad (1)\end{aligned}$$

Similarly, let $\mathbf{w}_i$ be the weight vector for task i. Then

$$\begin{aligned}\alpha_i V_i^{\pi_R} &\leq \alpha_i V_i^{\pi_i} \qquad\qquad \forall\, i \in \mathcal{T}\\ \sum_{i \in \mathcal{T}} \alpha_i V_i^{\pi_R} &\leq \sum_{i \in \mathcal{T}} \alpha_i V_i^{\pi_i}\\ \sum_{i \in \mathcal{T}} \alpha_i [\Lambda^{\pi_R}]\mathbf{w}_i &\leq \sum_{i \in \mathcal{T}} \alpha_i V_i^{\pi_i}\\ [\Lambda^{\pi_R}] \sum_{i \in \mathcal{T}} \alpha_i \mathbf{w}_i &\leq \sum_{i \in \mathcal{T}} \alpha_i V_i^{\pi_i}\\ [\Lambda^{\pi_R}]\mathbf{w}_R &\leq \sum_{i \in \mathcal{T}} \alpha_i V_i^{\pi_i}\\ V_R^{\pi_R} &\leq \sum_{i \in \mathcal{T}} \alpha_i V_i^{\pi_i} \qquad (2)\end{aligned}$$

Combining inequalities 1 and 2 gives us

$$\frac{1}{\alpha} \sum_{i \in \mathcal{T}} \alpha_i V_R^{\pi_i} \leq V_R^{\pi_R} \leq \sum_{i \in \mathcal{T}} \alpha_i V_i^{\pi_i} \qquad (3)$$

We note that, when considered point-wise over the states, the left hand side can be no greater than the max over all terms in the sum and this max cannot exceed the value under the optimal policy. Using (s) to indicate indexing into a matrix or vector by the state s, it follows that

$$\forall s \; \max_{i \in \mathcal{T}} V_R^{\pi_i}(s) \leq V_R^{\pi_R}(s) \leq \sum_{i \in \mathcal{T}} \alpha_i V_i^{\pi_i}(s) \qquad (4)$$

Finally, consider the successor feature approximation $\tilde{\Lambda}^{\pi_i}$. Then $\forall\, i, j \in \mathcal{T}$ and $\forall\, s \in S$,

$$\begin{aligned}V_j^{\pi_i}(s) &= \Lambda^{\pi_i}(s)^T \mathbf{w}_j\\ &= \left[\tilde{\Lambda}^{\pi_i}(s) - \Delta^{\pi_i}(s)\right]^T \mathbf{w}_j\end{aligned}$$

And it follows that

$$\begin{aligned}\tilde{\Lambda}^{\pi_i}(s)^T \mathbf{w}_j - |\Delta^{\pi_i}(s)^T \mathbf{w}_j| \leq \\ \left[\tilde{\Lambda}^{\pi_i}(s) - \Delta^{\pi_i}(s)\right]^T \mathbf{w}_j \leq \\ \tilde{\Lambda}^{\pi_i}(s)^T \mathbf{w}_j + |\Delta^{\pi_i}(s)^T \mathbf{w}_j|\end{aligned}$$

And thus that

$$\begin{aligned}\tilde{V}_j^{\pi_i}(s) - \boldsymbol{\epsilon}_j^{\pi_i}(s) &\leq \\ V_j^{\pi_i}(s) &\leq \\ \tilde{V}_j^{\pi_i}(s) + \boldsymbol{\epsilon}_j^{\pi_i}(s) & \qquad (5)\end{aligned}$$

Combine this information with inequality 4 and have $\forall\ s$,

$$\begin{aligned}\max_{i\in\mathcal{T}}\left[\tilde{V}_R^{\pi_i}(s) - \boldsymbol{\epsilon}_R^{\pi_i}(s)\right] &\leq \\ V_R^{\pi_R}(s) &\leq \\ \sum_{i\in\mathcal{T}}\alpha_i\tilde{V}_i^{\pi_i}(s) + \sum_{i\in\mathcal{T}}\alpha_i\boldsymbol{\epsilon}_i^{\pi_i}(s) & \qquad (6)\end{aligned}$$

□

B. Tightness of Barreto et al. Bounds

We consider the tightness of the Barreto et al. (2017) bounds in the case no approximation error, i.e., $\epsilon = 0$. Their bounds use only $\phi_{\max}$, and the norm of the difference between reward function weight vectors. Given this limited amount of information, these bounds are close to tight. These results do not preclude obtaining tighter bounds using more information, as we and others do.

B.1. Lower Bound

Consider an MDP with just one possible policy, π. For all i and all j in τ, $\pi_j^* = \pi$, and for all s and a, $Q_i^{\pi_j^*}(s,a) = Q_i^{\pi_i^*}(s,a)$, which shows that Theorem 1 from Barreto et al. holds with equality in this case.

B.2. Upper Bound

Consider and MDP with two states s_0, s_1, and two actions, a_0 and a_1. In any case, $P(s_k|s_i,a_k) = 1.0,\ i,k \in \{0,1\}$. Define reward features over states with $\phi_i(s_i) = \delta(s_i)$, $i \in \{0,1\}$, i.e., delta functions on the state. $\phi_{\max} = \max_i \|\phi_i\| = 1$. The reward function is, thus, parameterized by $\boldsymbol{w} = [w_1, w_2]$. For $\boldsymbol{w}_1 = [1,0]$, π_1^* chooses action 0 in all states with:

$$\begin{aligned}Q_1^{\pi_1^*}(s_0,a_0) &= \frac{w_1}{1-\gamma} = \frac{1}{1-\gamma}\\ Q_1^{\pi_1^*}(s_1,a_0) &= \frac{\gamma w_1}{1-\gamma} = \frac{\gamma}{1-\gamma}\\ Q_1^{\pi_1^*}(s_0,a_1) &= \frac{\gamma^2 w_1}{1-\gamma} = \frac{\gamma^2}{1-\gamma}\\ Q_1^{\pi_1^*}(s_1,a_1) &= \frac{\gamma^2 w_1}{1-\gamma} = \frac{\gamma^2}{1-\gamma}\end{aligned}$$

For $\boldsymbol{w}_2 = [w_1, w_2] = [0,1]$, the optimal policy chooses action 1 in all states with:

$$\begin{aligned}Q_2^{\pi_2^*}(s_0,a_0) &= \frac{\gamma^2 w_2}{1-\gamma} = \frac{\gamma^2}{1-\gamma}\\ Q_2^{\pi_2^*}(s_1,a_0) &= \frac{\gamma^2 w_2}{1-\gamma} = \frac{\gamma^2}{1-\gamma}\\ Q_2^{\pi_2^*}(s_0,a_1) &= \frac{\gamma w_2}{1-\gamma} = \frac{\gamma}{1-\gamma}\\ Q_2^{\pi_2^*}(s_1,a_1) &= \frac{w_2}{1-\gamma} = \frac{1}{1-\gamma}\end{aligned}$$

The value of applying π_1^* on task 2 is:

$$\begin{aligned}Q_2^{\pi_1^*}(s_0,a_0) &= \frac{w_1}{1-\gamma} = 0\\ Q_2^{\pi_1^*}(s_1,a_0) &= \frac{\gamma w_1}{1-\gamma} = 0\\ Q_2^{\pi_1^*}(s_0,a_1) &= \frac{\gamma^2 w_1}{1-\gamma} = 0\\ Q_2^{\pi_1^*}(s_1,a_1) &= \frac{\gamma^2 w_1}{1-\gamma} = 0\end{aligned}$$

The suboptimality of using this policy in state 1 is:

$$Q_2^{\pi_2^*}(s_1,a_1) - Q_2^{\pi_1^*}(s_1,a_1) = \frac{1}{1-\gamma}.$$

From the Barreto et al. bound for $\epsilon = 0$, we have:

$$\begin{aligned}Q_2^{\pi_2^*}(s_1,a_1) - Q_2^{\pi_1^*}(s_1,a_1) &\leq \frac{2}{1-\gamma}\phi_{\max}\|\boldsymbol{w}_1 - \boldsymbol{w}_2\|\\ &= \frac{2}{1-\gamma}\end{aligned}$$

This shows that the Barreto et al. bound is close to tight – it cannot be improved by more than a factor of 2 without taking additional information into account.

C. Hunt et al. Theorem Applied to Hard-Max

An upper bound on the optimal action-value function can be constructed based on Theorem 3.2 of Hunt et al. (2019) by removing their C_b^∞ term, since it is subtracted away to get the optimal Q function for the new task. This term corrects for the divergence between the known policies and the optimal policy for the new task.

Due to the fact that soft Q-learning approaches hard-max Q-learning as the temperature parameter $\alpha \to 0^+$, it is intuitive to think that this result applies to hard-max Q-functions as well. However, while our result shows that such a bound does indeed hold, it does not follow from their Theorem 3.2, which we show with a proof by contradiction.

Proof. Assume that Theorem 3.2 applies to hard-max Q-functions. In that case, we have optimal hard-max policies π_i, π_j with respect to reward functions r_i, r_j, corresponding action-value functions Q^i, Q^j, a composite reward function $r_b \equiv br_i + (1-b)r_j$, and the following equation from Theorem 3.2 holds:

$$Q_b^*(s,a) = bQ^i(s,a) + (1-b)Q^j(s,a) - C_b^\infty(s,a) \quad (7)$$

Since this must hold for all bounded action-value functions, we consider a case with reward functions $\hat{r}_i \equiv kr_i$ and $\hat{r}_j \equiv kr_j$, where $k > 0$ is a bounded real-valued constant and $k \neq 1$. We therefore also have a new, different composite reward function defined by the weight b: $\hat{r}_b \equiv b\hat{r}_i + (1-b)\hat{r}_j \equiv kr_b$. Scaling a reward function by a positive constant does not change the corresponding optimal hard-max policies but does scale the action-value functions for those policies by the same constant, so we have $\hat{\pi}_i \equiv \pi_i$, $\hat{\pi}_j \equiv \pi_j$, $\hat{Q}^i \equiv kQ^i$, $\hat{Q}^j \equiv kQ^j$, and $\hat{Q}_b^* \equiv kQ_b^*$. As $C_b^\infty(s,a)$ is a function of the policies and not the reward or value functions, it follows that this function also does not change with our scaling. Therefore, the following holds according to Theorem 3.2:

$$\hat{Q}_b^*(s,a) = b\hat{Q}^i(s,a) + (1-b)\hat{Q}^j(s,a) - C_b^\infty(s,a) \quad (8)$$

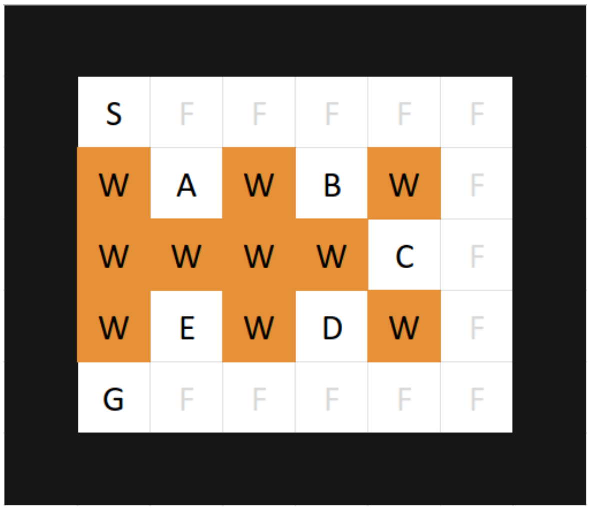
Let us further stipulate that r_i and r_j are different enough such that $C_b^\infty(s,a) > 0$ for at least one state-action pair, i.e., there is some divergence between the policies. For any such pair, we can rearrange Equations 7 and 8 as follows:

$$\begin{aligned}
&bQ^i(s,a) + (1-b)Q^j(s,a) - Q_b^*(s,a) \\
&\qquad = C_b^\infty(s,a) \\
&\qquad = b\hat{Q}^i(s,a) + (1-b)\hat{Q}^j(s,a) - \hat{Q}_b^*(s,a) \\
&bQ^i(s,a) + (1-b)Q^j(s,a) - Q_b^*(s,a) \\
&\qquad = b\hat{Q}^i(s,a) + (1-b)\hat{Q}^j(s,a) - \hat{Q}_b^*(s,a) \\
&bQ^i(s,a) + (1-b)Q^j(s,a) - Q_b^*(s,a) \\
&\qquad = bkQ^i(s,a) + (1-b)kQ^j(s,a) - kQ_b^*(s,a) \\
&bQ^i(s,a) + (1-b)Q^j(s,a) - Q_b^*(s,a) \\
&\qquad = k\big[bQ^i(s,a) + (1-b)Q^j(s,a) - Q_b^*(s,a)\big] \\
&\quad k = 1
\end{aligned}$$

However, this contradicts our requirement that $k \neq 1$ and thus Theorem 3.2 cannot hold. □

D. Additional Experimental Results

We performed additional experiments with the same underlying environment as Gridworld, but with the alternate grid shown in Figure 5, which we reference as Gridworld 5x6. For this grid, picking up any object requires increasing the number of steps taken to reach the goal, so depending on the reward assigned to a given object, it may or may not be optimal leave the shortest path to the goal to pick it up.



S	F	F	F	F	F
W	A	W	B	W	F
W	W	W	W	C	F
W	E	W	D	W	F
G	F	F	F	F	F

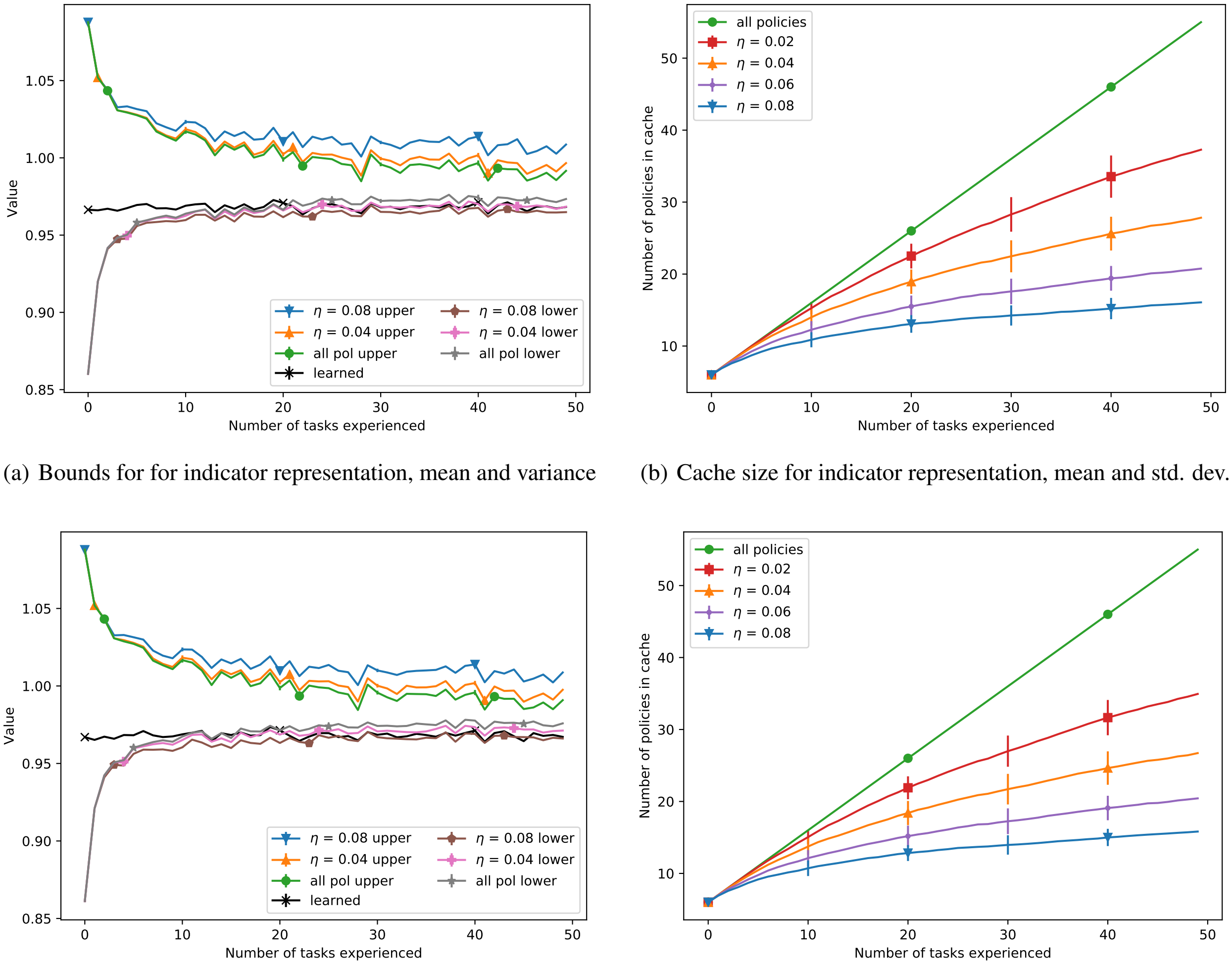
Figure 5. GridWorld 5x6 with start (S), walls (W), goal (G), objects (A-E)

There are three additional significant differences from the Gridworld experiments in Section 5: (1) new tasks were convex combinations of the base tasks sampled from a uniform, random distribution over such combinations, (2) feature-based representations were used, and (3) function approximation was used for the successor features in the form of neural networks.

For the five base tasks in these experiments, a reward of 1.0 is received for reaching the goal and a reward of 0.4 is received for picking up one of the objects. In each run, 50 additional tasks were sampled from a uniform, random distribution over convex combinations of the base tasks. We collected data from 100 runs, each of which corresponds to a different seed for the sampling of tasks.

The first state representation we used consisted of a one-hot encoding of the agent's position concatenated with the current object inventory, which has an indicator for each object, and a constant bias feature. The second representation was pixel-based with 50x60 pixels (10x10 for each cell) and 3 color channels where the agent and objects were visualized as circles of different colors.

The results for these experiments are shown in Figure 6, which follow a similar pattern to those in Section 5. These results demonstrate that the benefits of our method appear even when exact representations are not used. The differences between the results for the two representations are the effect of a slightly larger error in the SF approximations for the pixel-based representation.



(a) Bounds for for indicator representation, mean and variance

(b) Cache size for indicator representation, mean and std. dev.

(c) Bounds for pixel representation, mean and variance

(d) Cache size for pixel representation, mean and std. dev.

Figure 6. Data for the GridWorld 5x6 environment, 100 runs

E. Experimental Methodology

E.1. Gridworld 5x6

For the Gridworld 5x6 experiments in Section D, multiple neural network architectures and hyperparameter settings were explored. Table 1 lists the necessary hyperparameters, the values used for the reported results, and the set of values considered during our experiments. The values used for our results were chosen based on preliminary experiments which provided an estimate of the approximation error of the successor features after training as well as the performance of the induced policy.

For the indicator-based state representation, MLPs with different numbers of layers with ReLU nonlinearities were considered, while for the image-based representation, the structure used was that of DQN (Mnih et al., 2015) with three convolutional layers and two fully-connected layers. For each task, SFs were trained using a modified version of SFQL. The modifications allowed for the use of arbitrary neural network models, replay memory, batch sizes larger than one, and a target network. Each task was trained on independently, so GPI was not used.

E.2. Reacher

The hyperparameters for the Reacher environment are shown in Table 2.

References

Barreto, A., Dabney, W., Munos, R., Hunt, J. J., Schaul, T., van Hasselt, H. P., and Silver, D. Successor features for transfer in reinforcement learning. In *Advances in*

Hyperparameter	Value Used For Results	Other Values Considered
Learning Rate	0.1	0.01, 0.05
Minibatch Size	32	1, 16
Optimization Frequency	1	32
Target Update Frequency	500	1, 32
Optimizer	SGD	SGD
ϵ decay schedule	linear	constant
ϵ start	1.0	1.0
ϵ end	0.05	0.15
ϵ decay period (# samples)	500K	100K
ϵ (for constant)	N/A	0.15
# Training samples	500K	200K
Indicator-based network arch	2-layer	linear, 3-layer
Indicator-based hidden layer width	72	36
Image-based Network architecture	DQN structure (Mnih et al., 2015)	N/A

Table 1. Hyperparameters for the GridWorld 5x6 experiments. Frequencies refer to the number of samples collected between steps.

Hyperparameter	Value Used For Results	Other Values Considered
Learning Rate	1e-5	1e-3, 1e-4
Minibatch Size	128	32
Optimization Frequency	1	N/A
Target Update Frequency	500	N/A
Optimizer	Adam	SGD
ϵ decay schedule	constant	linear
ϵ start	N/A	1.0
ϵ end	N/A	0.05
ϵ decay period (# samples)	N/A	100K, 500K
ϵ (for constant)	0.1	0.15
# Training samples	15M	1M, 5M, 10M
Network arch	2-layer	linear, 3-layer
Hidden layer width	256	64, 128

Table 2. Hyperparameters for the Reacher experiments. Frequencies refer to the number of samples collected between steps.

Neural Information Processing Systems 30, pp. 4055–4065. Curran Associates, Inc., 2017.

Hunt, J., Barreto, A., Lillicrap, T., and Heess, N. Composing entropic policies using divergence correction. In Chaudhuri, K. and Salakhutdinov, R. (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2911–2920. PMLR, 2019.

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. Human-level control through deep reinforcement learning. *Nature*, 518(7540): 529–533, February 2015. ISSN 00280836.