NurSpecialist: Duel-Agent Reinforcement Learning for Dynamic Hospitalised Intervention Regimes using Electronic Health Records

Zhiyao Luo,[1] Peter Watkinson,[1] Tingting Zhu,[1].

[1]University of Oxford

zhiyao.luo@eng.ox.ac.uk, peter.watkinson@ndcn.ox.ac.uk, tingting.zhu@eng.ox.ac.uk

Abstract

Reinforcement Learning (RL) is a promising approach to provide decision support to assist clinicians in intervention decision-making as electronic patient data are now routinely acquired. As an analogy to nurses and specialists/consultants working cooperatively in a hospital setting, we propose a duel-agent offline RL framework, *NurSpecialist*, where a *nurse* agent conducts regular checks on a patient's health status and works with a *specialist* agent that decides on the treatment regimes. Our framework considers offline multi-agent cooperative learning and proposes an inter-agent communication mechanism. Our framework outperformed classical RL algorithms and was validated on two intervention regimes in the intensive care setting with over 17,000 patients.

1 Introduction

Intervention treatment is critical and can directly affect patients' clinical outcomes in hospitalised settings. However, the optimal types of intervention and dosing can vary from patient to patient and doctor to doctor; therefore, there is yet a gold standard for optimal intervention. Dynamic treatment regime aims to tackle this issue by considering the individual patients' ongoing and past responses to treatments.

A dynamic treatment regime (DTR) system comprises a sequence of medical actions in the decision-making process to inform treatment strategies. With technological advances, we can now collect treatment regime data electronically. Hence, it provides the opportunity to understand the progression of complex diseases via courses of treatment using reinforcement learning (RL)[Gottesman et al. 2019]. RL has demonstrated its potential in solving dynamic intervention regimes in many hospitalised scenarios such as glucose management in diabetes mellitus[Tejedor, Woldaregay, and Godtliebsen 2020], mechanical ventilation weaning[Prasad et al. 2017], and drug dosage control for sepsis patients[Raghu et al. 2017a] [Raghu et al. 2017b] [Saria 2018].

RL-based DTR faces significant challenges when dealing with electronic health record (EHR) data. While RL attempts to optimise the sequence of decisions natively by interacting with the environment, online exploration is inherently difficult to conduct as EHR data are generally collected retrospectively. In this study, we focus on offline RL, where the agent learns from a fixed set of human doctors' treatment decision trajectories in the hospital. Offline RL does not require interaction with the environment and thus eliminates the risk of applying faulty treatment policies (via exploration) to patients during training. Offline RL for DTR has its unique challenges:

1) **Outcome Imbalance and Data Sparsity** Treatment trajectories are imbalanced: (i) it is, therefore, difficult to model deceased versus survived treatment trajectories with highly imbalanced classes; (ii) patients are prescribed with similar treatment strategies despite having different demographic, phenotypes and diseases; (iii) some treatments are rarely used, or they are specific to a small group of patients only.
2) **Intervention-Aware Action Repetition** Medical interventions and medication prescriptions do not change frequently: (i) drugs with a fixed dosage usually require hours to take effect; (ii) interventions such as endotracheal tube mechanical ventilation are invasive and inappropriate to insert and wean frequently.
3) **Explainable Assignment of Treatment Strategy** changes of treatment may be infrequent whereas important. It is crucial to learn *when* to change treatment and *which* treatment to recommend, such that algorithms can help clinicians understand how a treatment leads to a particular outcome.

To address the above challenges in DTR, we propose a two-agent framework, *NurSpecialist*, mimicking the responsibility of a human nurse and specialist, respectively. A *nurse* agent acts as an early-warning system to alert the *specialist* when the patient is deteriorating or in need of a change of the current treatment strategy, and the *specialist* agent prescribes necessary treatments to the patient. The *nurse* monitors the patient's health regularly, and a request is made to the *specialist* only when a suspected change in the current treatment is needed.

In the proposed framework, the *nurse* agent reduces duplicated state-action pairs, which are inherently too large to optimise, prior to sending to the *specialist* agent, establishing a more balanced data learning. Besides, the presence of a *nurse* agent makes strategy assignment more transparent

by presenting the probability of changing treatments. It also helps to reduce unnecessary treatment changes.

2 Preliminary

Dynamic Treatment Regime (DTR) is formulated as a sequential decision process, where an *episode* starts from admission and terminates if the patient is discharged or deceased in a hospitalised setting.

Given an observed dataset $\mathcal{D} = \{(\boldsymbol{o}_t^i, a_t^i, r_t^i\}_{t=1}^{l_n})_{i=1}^N$, where it contains N admission treatment trajectories, each with a length up to l_n. At time-stamp t of the i^{th} trajectory, we define an observation set $\boldsymbol{o}_t^i \in \mathbb{R}^d$ with d_{obs} dimensions, human doctor's action a_t^i, and the corresponding reward r_t^i. We assume that all human doctors make decisions following an identical policy distribution $\beta : \mathcal{S} \to \mathcal{A}$ with full exploitation (See Appendix). Each patient admission is considered an independent episode containing the full journey of a hospital stay. In this work, we use the terms *action* and *treatment* interchangeably.

We model an offline episode as a Markov Decision Process (MDP) with $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, p, R, \gamma\}$, where $\mathcal{A} = \{a_m\}_{m=1}^M$ is the discrete action space with M actions, $\mathcal{S} \in \mathbb{R}^{d_s}$ is the state space with d_s dimension, $p(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, a_t)$ is the state transition function that maps transition from the current state $\boldsymbol{s}_t$ to the next state $\boldsymbol{s}_{t+1}$ following an action a_t at time t. $R(\boldsymbol{s}_t, a_t) : S \to \mathbb{R}$ is the reward function, and $\gamma \in (0, 1)$ is the discounted factor. The goal of the RL agent is to find an optimal policy $\pi : \mathcal{S} \to \mathcal{A}$ that can maximise the expected reward $G(\boldsymbol{s}_t) = \mathbb{E}_\pi[\sum_t \gamma^{t-1} R(\boldsymbol{s}_t, a_t)]$.

3 Data Cohort and preprocessing

Our datasets (see Appendix) are curated from MIMIC-III [Johnson et al. 2016], an open-source database comprising de-identified health data associated with patients in the critical care unit (ICU). To validate our methodology, we select two commonly-used hospitalised DTR tasks: oxygen therapy and vasopressor-intravenous dosage regime for sepsis treatment.

Oxygen Therapy Cohort in General ICU

This dataset contains 48 hours of EHR data on over 7,000 adult patients admitted for various reasons, such as cardiac, surgical, and trauma. In addition to the data curation process in [Goldberger et al. 2000], we exclude patients with no ventilation or those with a hospital longer stay longer than 30 days. The task is to train the agent to learn the correct time to apply/wean mechanical ventilation for the patients.

Sepsis Cohort with Vasopressor-Intravenous Regime

This dataset [Komorowski et al. 2018] contains a 10,000 sepsis patients targeting intravenous (IV) fluids and vasopressors dosing control in the ICU. In this task, an agent should learn to prescribe the optimal dosing of treatments and be aware of when to switch to a new combination of treatments according to the observed physiological changes of a patient.

4 Methodology

Augmented Markov Decision Process (MDP) for NurSpecialist

In hospital wards, a nurse is responsible for checking the patient regularly and evaluating his/her health status to decide whether to request the specialist/consultant to change the ongoing treatment. Here we augment the MDP into a two-agent setup: $\mathcal{M}_{sp} = \{\mathcal{S}, \mathcal{A}, p', R, \gamma\}$ and $\mathcal{M}_n = \{\mathcal{S}, \mathcal{A}_n, p', R, \gamma\}$. $\mathcal{M}_{sp}$ is the same MDP setup as described in Section Preliminary, except for a different state transition function p'. $\mathcal{M}_n$ is an augmented MDP having a separate action space $\mathcal{A}_n = \{k_0, k_1\}$, where $k_0 \doteq a_{t-1}$ denotes the action to *continue* using the same treatment as before, and $k_1 \doteq a_t$ denotes the action to *change* by requesting the *specialist* to prescribe a treatment.

We denote a collaborative decision made jointly by the *nurse* and *specialist* agents at time t as υ_t, and a collaborative policy as $\tau : \mathcal{S} \times \mathcal{A}_{sp} \times \mathcal{A}_n \to \mathcal{A}_{sp}$. By definition, collaborative action is decided solely upon *specialist* at the initial step/treatment and then is decided jointly by the *nurse* and the *specialist* agents. The collaborative action υ_t can be defined as

$$\upsilon_t \doteq \begin{cases} a_t \sim \pi_{sp} & \text{if } t = 0, \\ \upsilon_m | \upsilon_{t-1} = \upsilon_m & \text{if } t > 0 \text{ and } k_t = k_0, \\ a_t \sim \pi_{sp} & \text{if } t > 0 \text{ and } k_t = k_1. \end{cases} \quad (1)$$

where υ_{t-1} refers to the collaborative action made in the $t-1$ time step, and $\upsilon_m \in \mathcal{A}$ is a specific action which belongs to the *specialist*'s action space. It is expected that *NurSpecialist* follows the previous action if the *nurse* agent decides not to request *specialist*. Otherwise, it follows the *specialist* 's decision.

By combining the two MDPs $\mathcal{M}_{sp}$ and $\mathcal{M}_n$, we extend the original MDP to a fully cooperative multi-agent reinforcement learning (MARL) framework with a *nurse* agent and a *specialist* agent. In an offline setup, we only train the model with human doctors' labels of action sampled from a retrospective dataset. Therefore the transition function is

$$p'(s_{t+1}|\boldsymbol{s}_t, a_n) = p(s_{t+1}|\boldsymbol{s}_t, a, \beta_{sp}, \beta_n), \quad (2)$$

where the *nurse* 's policy from human doctors is obtained by comparing if the current *specialist* 's action is the same as the previous step. Since β_n is induced by β_{sp}, $\mathcal{M}_{sp}$ and $\mathcal{M}_n$ are two independent MDPs when the *specialist* is stationary. The optimal Bellman equation for independent Q learning still holds by learning from offline human doctors' policies β_{sp} and its induced policy β_n. It also indicates that no fundamental parameter update change shall be made to adapt to any value-based single-agent algorithms. This property provides great convenience for implementing *NurSpecialist* in different offline RL scenarios.

Neural Network Realization of *NurSpecialist*

State representation learning. During learning DTR, the RL agents receive a set of observational features that implicitly describe the state. At each time step t, the agent receives the current observation alongside the recent history. We denote

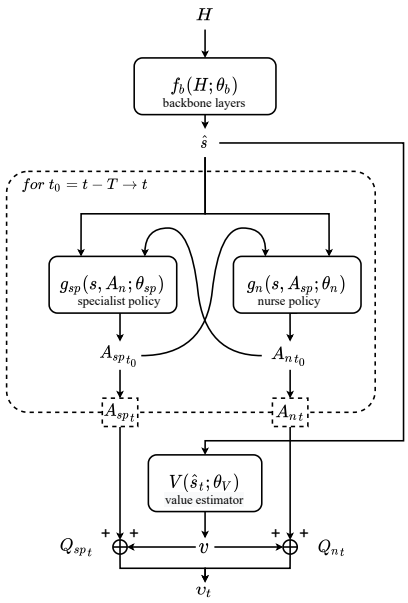

Figure 1: **The graphical representation of the *NurSpecialist* framework**: state representation $\hat{s}$ is learnt from an observation window processed by backbone model f_b, pretrained by auxiliary tasks. The *nurse* and *specialist* agents take $\hat{s}$ as input and produce their respective advantages A_{sp} and A_n. The *nurse* and *specialist* agents share a centralised value estimator V. A collaborate action is made considering both policies by communicating with each other for T times, where T is the observation window length.

the observation of a given time interval as $\boldsymbol{H}_{t-T:t}$, where T is the observation window. We use a LSTM-based backbone model $f_b(\mathtt{H}_{t-T:t}; \theta_b)$ to learn a representation of state $\hat{s}$.

The backbone model is firstly trained on auxiliary tasks for a more generalisable state representation [Bellemare et al. 2019] before being used for reinforcement learning. We choose the remaining length of stay (LOS) as the auxiliary task for both oxygen therapy and sepsis tasks since the treatment effectiveness is directly correlated to the remaining survival time of hospitalised patients.

Multi-agent policy learning. The learnt state representation is inserted to two policy models $g_{sp}(\hat{\boldsymbol{s}}; \theta_{sp})$ and $g_n(\hat{\boldsymbol{s}}, \boldsymbol{a}_{t-1}; \theta_n)$ for approximating Q value of *nurse* and *specialist*, respectively (see Fig **??**). Apart from the (learnt) state, the *nurse* agent takes the *specialist* 's action from the previous step as an extra input since the *nurse* policy relies on the treatment previously applied to the patient. Here $\mathtt{a}_{t-T:t}$ is the stacked one-hot encoding vector of the previous actions. Inspired by Wang et al. [2016] 's Dueling architecture, the advantage function $A_{sp}(s, a)$ and $A_n(s, a)$ are used as input to another agent's model for mutual communication.

Communication between agents. For the nurse-specialist interaction structure, communication means exchanging decisions iteratively. We therefore implement the following: the network of *nurse* and *specialist* are $g_n(s, A_{sp_{t-1}}; \theta_n)$ and $g_{sp}(s, A_{n_{t-1}}; \theta_n)$, where A_n and A_{sp} are the advantage function of the *nurse* and *specialist*, respectively, and an value function estimator $V(s; \theta_V) : \mathcal{S} \to \mathbb{R}$ is added. Note that the value function is shared between the *nurse* agent and the *specialist* agent(See Appendix for explanation). The architecture is explained in Algorithm 1.

Algorithm 1: *nurse-specialist* Sequential Communication

function MAIN_FORWARD($\mathtt{H_T}$)
  $\hat{\boldsymbol{s}}_{t-T:t} = f_b(\mathtt{H_T}; \theta_b)$
  $v = V(\hat{s}_t; \theta_V)$
  $A_{nt}, A_{sp_t} =$ COMMUNICATION_FORWARD($\boldsymbol{s}_{t-T:t}$)
  $Q_n \leftarrow v + A_{nT}$
  $Q_{sp} \leftarrow v + A_{spT}$
  output: Q_n, Q_{sp}
end function

function COMMUNICATION_FORWARD($\boldsymbol{s}_{t-T:t}$)
  initialize RNN hidden state $h_n = 0, h_{sp} = 0$
  initialize $A_{sp} = 0^{\|\mathcal{A}\| \times 1}$
  for $t_0 = t - T \to t$ **do**
    $A_n = g'_n(s_{t_0}, A_{sp}; \theta_n)$
    $A_n \leftarrow A_n - \frac{1}{\|\mathcal{A}_n\|} \sum_{a^n} A_n$
    $A_{sp} = g'_{sp}(s_{t_0}, A_n; \theta_{sp})$
    $A_{sp} \leftarrow A_{sp} - \frac{1}{\|\mathcal{A}_{sp}\|} \sum_{a^{sp}} A_{sp}$
  end for
  output: A_n, A_{sp}
end function

Loss Optimization

***Nurse-specialist* Consistency.** Since the *nurse* MDP is induced from the *specialist* MDP, the *nurse* agent and *specialist* agent have the same value function and underlying Q function (See Appendix). Therefore, the Q value learnt by the *nurse* should be identical to the Q value learnt by the *specialist* corresponding to $\mathcal{A}_n \mapsto \mathcal{A}$. To achieve this, a consistency loss is applied as

$$L_\phi = \boldsymbol{m}_r L_\psi + \neg \boldsymbol{m}_r L_\omega \tag{3}$$

where $\neg$ is the logical negation symbol, $\boldsymbol{m}_\psi \in \mathbb{R}^{1 \times b}$ for $\forall \boldsymbol{m}_i \in \{0, 1\}$ is a mask vector over batch b, denoting if a request is made, and L_ψ, L_ω are the respective mean square error (MSE) when the *nurse* agent made a *request* or *continue* decision:

$$\begin{cases} L_\omega = MSE(Q_n(\mathtt{H}, k_0) - Q_{sp}(\mathtt{H}, a_{t-1})) \\ L_\psi = MSE\Big(Q_n(\mathtt{H}, k_1) - Q_{sp}(\mathtt{H}, argmax_{a \in \mathcal{A}_{sp}} Q_{sp_t}(\mathtt{H}, a))\Big) \end{cases} \tag{4}$$

Mis-communication in *Nurse-specialist*. We identify two types of miscommunication between *nurse* and *specialist* agent: *missed request* and *unnecessary request*. *Missed request* is defined when the *nurse* does not request the *specialist* while the *specialist* prescribes a treatment which is different to the previous treatment. *Unnecessary request* is when the *nurse* requests the *specialist*, but the *specialist* makes no change to the treatment. Consider a miscommunication mask $\boldsymbol{m}_\eta \in \mathbb{R}^{1 \times b}$ for $\forall \boldsymbol{m}_i \in \{0, 1\}$,

$$\boldsymbol{m}_\eta = \delta(\delta(k_m = k_0 \wedge a_t \neq v_{t-1}) \vee \delta(k_m = k_1 \wedge a_t = v_{t-1})). \tag{5}$$

We borrow the idea from conservative Q learning to penalise the model for producing the miscommunicated joint

Table 1: Off-Policy Evaluation on Oxygen Therapy and Sepsis Cohorts

$\hat{v}^{\pi}_{WIS}$	Oxygen Therapy Cohort DQN	Oxygen Therapy Cohort DDQN	Sepsis Cohort DQN	Sepsis Cohort DDQN
π_{NS}	42.61 ± 18.42	35.21 ± 18.42	$\mathbf{67.12 \pm 5.07}$	$\mathbf{68.95 \pm 7.18}$
$\pi_{NS_{sp}}$	$\mathbf{44.35 \pm 16.15}$	$\mathbf{38.28 \pm 15.36}$	23.63 ± 5.52	25.44 ± 6.15
π_{NS_r}	14.71 ± 13.21	14.86 ± 13.15	21.36 ± 4.26	22.78 ± 7.48
π_{NS_0}	18.94 ± 4.49	18.95 ± 4.51	24.17 ± 4.32	25.05 ± 5.24
π_{SA}	37.67 ± 10.43	31.02 ± 4.75	63.95 ± 11.14	59.53 ± 8.84
DDPG	23.16 ± 3.06		28.68 ± 2.67	
CQL	30.41 ± 0.70		30.30 ± 2.81	
β	64.21		45.66	
π_r	29.09		-9.74	
π_0	38.49		21.73	

decision. Empirically, we applied a penalty to *nurse* 's miscommunication by suppressing its' Q value. The miscommunication loss is given by

$$L_\zeta = MSE\left(\boldsymbol{m}_{mis}(Q_n(\mathtt{H}, k_0) - Q_n(\mathtt{H}, k_1))^-\right), \quad (6)$$

where $\{k_0, k_1\} = \mathcal{A}_n$ denoting the mis-communicated action is a *continue* action or *request* action, respectively.

Overall, the *NurSpecialist* parameters are updated by

$$\theta \leftarrow \theta - \nabla[L_n + L_{sp} + w_1 L_\phi + w_2 L_\zeta]. \quad (7)$$

where w_1 and w_2 are hyper-parameters to be optimised during training.

5 Evaluation

In this section, we introduce off-policy evaluation to compare the learning performance between a single-agent RL and the *NurSpeciliast*. For simplicity, we denote *NurSpecialist* as *NS* and single-agent policy as *SA*, respectively. We use weighted importance sampling (WIS), a Monte-Carlo technique for estimating the expectation of a policy by providing samples from a different policy (such as the human doctor's policy). A brief description of WIS can be found in Appendix.

We denote the human doctor's policy as β, a random policy as π_r and a zero-drug policy as π_0. The WIS score of the human doctor's policy (denoted as β) is calculated by taking the mean of discounted rewards across the testing set. WIS scores for other policies are estimated using Eq 20. We also investigate how *nurse* would affect the treatment decision of *specialist* in three scenarios: (i) the *nurse-specialist* communication is disabled by removing the *nurse* agent completely, i.e., $\pi_{NS_{sp}}$; (ii) the *nurse* agent is replaced with a random policy, i.e., π_{NS_r}; and (iii) the *nurse* agent always decides to continue using the last treatment, i.e., π_{NS_0}.

RL Benchmarks. We implemented two standard off-policy algorithms, i.e., DQN, DDQN; an offline algorithm CQL[Kumar et al. 2020]; and an off-policy actor-critic algorithm DDPG[Silver et al. 2014] as baselines. Since DDPG is natively incapable of discrete action space, we follow the discretisation trick proposed in [Lowe et al. 2017] to compute DDPG on both tasks.

Results Table 1 shows the estimated reward for each policy on the two datasets. Among single-agent policies, DQN achieves the highest estimated return in the oxygen therapy cohort, while DDQN moderately outperforms DDQN in the sepsis dataset. DDPG and CQL fail to learn a good policy, given that their estimated returns are worse than a random policy or a "doing nothing" policy that always continues using the last treatment. DDPG results in the poorest performance since it is inherently designed for continuous rather than discrete action space. Empirically, the discretisation trick of DDPG does not fit the two treatment datasets.

NS achieves the highest estimated return for both datasets, although NS with $\pi_{NS_{sp}}$ performs slightly better than NS with π_{NS} in the oxygen therapy cohort. Since the task of oxygen therapy decision is inherently simple (i.e., binary, to apply mechanical ventilation or not), having a *nurse* agent does not help to infer a better policy. However, all NS-based frameworks outperform SA and benchmarking algorithms, demonstrating that incorporating the *nurse* agent helps the *specialist* to learn during training. In the sepsis cohort, the NS outperforms all other NS-based policies. In particular, the *Specialist* agent alone (i.e., $\pi_{NS_{sp}}$) does not perform well due to the large action space of the sepsis dataset. In this situation, the *nurse* and *specialist* cooperate well and surpass the doctors' behavioural policy.

Discussion. Two simple reinforcement learning algorithms, i.e., DQN and DDQN, are used with the *NurSpecialist* structure and show extraordinary performance compared with single-agent baselines on real hospitalised data. When the action space is relatively small (e.g. in the oxygen therapy dataset), having dual-agent during the inference stage does not contribute to policy improvement significantly. However, the *nurse* agent is still necessary for the framework during training since our loss design and communication mechanism assists the *specialist* agent in learning a more robust policy. When dealing with larger action space, remaining treatments becomes dramatically important. In such cases, the *specialist* cannot prescribe treatments without the assistance of the *nurse* agent.

The *NurSpecialist* framework not only boosts policy performance but also helps clinicians to explain the learnt policy. The nurse agent produces the probability of changing treatment at each timestep, which can be interpreted as a credit assigner. Treatment change is usually linked to the deterioration/recovery of patients. Clinicians would observe a higher possibility from the *nurse* if there is a higher chance to change the treatment, potentially avoiding missing the best timing to change treatment plan based on patient's health status.

6 Conclusion

In this work, we introduced *NurSpecialist*, a multi-agent framework to learn when is the right time to change treatment (i.e., by the *nurse* agent) and type of treatments (i.e., by the *specialist* agent) fully offline. We described the unique challenges in DTR and discussed how these challenges negatively impact learning optimal policies for existing offline RL algorithms. *NurSpecialist* also provides more clinical insights than single-agent learning. In addition to recommending treatments, *NurSpecialist* serves as an early warning system for clinical decision support.

References

Bellemare, M.; Dabney, W.; Dadashi, R.; Ali Taiga, A.; Castro, P. S.; Le Roux, N.; Schuurmans, D.; Lattimore, T.; and Lyle, C. 2019. A geometric perspective on optimal representations for reinforcement learning. *Advances in neural information processing systems*, 32.

Esteban, A.; Frutos-Vivar, F.; Muriel, A.; Ferguson, N. D.; Peñuelas, O.; Abraira, V.; Raymondos, K.; Rios, F.; Nin, N.; Apezteguía, C.; et al. 2013. Evolution of mortality over time in patients receiving mechanical ventilation. *American journal of respiratory and critical care medicine*, 188(2): 220–230.

Fatemi, M.; Killian, T. W.; Subramanian, J.; and Ghassemi, M. 2021. Medical Dead-ends and Learning to Identify High-risk States and Treatments. *Advances in Neural Information Processing Systems*, 34.

Fu, J.; Kumar, A.; Nachum, O.; Tucker, G.; and Levine, S. 2020. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*.

Fujimoto, S.; Conti, E.; Ghavamzadeh, M.; and Pineau, J. 2019. Benchmarking batch deep reinforcement learning algorithms. *arXiv preprint arXiv:1910.01708*.

Fujimoto, S.; Meger, D.; and Precup, D. 2019. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, 2052–2062. PMLR.

Goldberger, A. L.; Amaral, L. A.; Glass, L.; Hausdorff, J. M.; Ivanov, P. C.; Mark, R. G.; Mietus, J. E.; Moody, G. B.; Peng, C.-K.; and Stanley, H. E. 2000. PhysioBank, PhysioToolkit, and PhysioNet: components of a new research resource for complex physiologic signals. *circulation*, 101(23): e215–e220.

Gottesman, O.; Johansson, F.; Komorowski, M.; Faisal, A.; Sontag, D.; Doshi-Velez, F.; and Celi, L. A. 2019. Guidelines for reinforcement learning in healthcare. *Nature medicine*, 25(1): 16–18.

Guez, A.; Vincent, R. D.; Avoli, M.; and Pineau, J. 2008. Adaptive Treatment of Epilepsy via Batch-mode Reinforcement Learning. In *AAAI*, 1671–1678.

Harris, C. R.; Millman, K. J.; van der Walt, S. J.; Gommers, R.; Virtanen, P.; Cournapeau, D.; Wieser, E.; Taylor, J.; Berg, S.; Smith, N. J.; Kern, R.; Picus, M.; Hoyer, S.; van Kerkwijk, M. H.; Brett, M.; Haldane, A.; del Río, J. F.; Wiebe, M.; Peterson, P.; Gérard-Marchant, P.; Sheppard, K.; Reddy, T.; Weckesser, W.; Abbasi, H.; Gohlke, C.; and Oliphant, T. E. 2020. Array programming with NumPy. *Nature*, 585(7825): 357–362.

Johnson, A. E.; Pollard, T. J.; Shen, L.; Lehman, L.-w. H.; Feng, M.; Ghassemi, M.; Moody, B.; Szolovits, P.; Anthony Celi, L.; and Mark, R. G. 2016. MIMIC-III, a freely accessible critical care database. *Scientific data*, 3(1): 1–9.

Kester, L.; and Stoller, J. K. 1992. Ordering respiratory care services for hospitalized patients: practices of overuse and underuse. *Cleveland Clinic Journal of Medicine*, 59(6): 581–585.

Komorowski, M.; Celi, L. A.; Badawi, O.; Gordon, A. C.; and Faisal, A. A. 2018. The artificial intelligence clinician learns optimal treatment strategies for sepsis in intensive care. *Nature medicine*, 24(11): 1716–1720.

Kostrikov, I.; Nair, A.; and Levine, S. 2021. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*.

Kumar, A.; Fu, J.; Soh, M.; Tucker, G.; and Levine, S. 2019. Stabilizing off-policy q-learning via bootstrapping error reduction. *Advances in Neural Information Processing Systems*, 32.

Kumar, A.; Zhou, A.; Tucker, G.; and Levine, S. 2020. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33: 1179–1191.

Liu, Y.; Logan, B.; Liu, N.; Xu, Z.; Tang, J.; and Wang, Y. 2017. Deep reinforcement learning for dynamic treatment regimes on medical registry data. In *2017 IEEE International Conference on Healthcare Informatics (ICHI)*, 380–385. IEEE.

Lowe, R.; Wu, Y. I.; Tamar, A.; Harb, J.; Pieter Abbeel, O.; and Mordatch, I. 2017. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in neural information processing systems*, 30.

Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; and Lerer, A. 2017. Automatic differentiation in PyTorch.

Prasad, N.; Cheng, L.-F.; Chivers, C.; Draugelis, M.; and Engelhardt, B. E. 2017. A reinforcement learning approach to weaning of mechanical ventilation in intensive care units. *arXiv preprint arXiv:1704.06300*.

Raghu, A.; Komorowski, M.; Ahmed, I.; Celi, L.; Szolovits, P.; and Ghassemi, M. 2017a. Deep reinforcement learning for sepsis treatment. *arXiv preprint arXiv:1711.09602*.

Raghu, A.; Komorowski, M.; Celi, L. A.; Szolovits, P.; and Ghassemi, M. 2017b. Continuous state-space models for optimal sepsis treatment: a deep reinforcement learning approach. In *Machine Learning for Healthcare Conference*, 147–163. PMLR.

Saria, S. 2018. Individualized sepsis treatment using reinforcement learning. *Nature medicine*, 24(11): 1641–1642.

Silver, D.; Lever, G.; Heess, N.; Degris, T.; Wierstra, D.; and Riedmiller, M. 2014. Deterministic policy gradient algorithms. In *International conference on machine learning*, 387–395. PMLR.

Sonabend, A.; Lu, J.; Celi, L. A.; Cai, T.; and Szolovits, P. 2020. Expert-supervised reinforcement learning for offline policy learning and evaluation. *Advances in Neural Information Processing Systems*, 33: 18967–18977.

Tejedor, M.; Woldaregay, A. Z.; and Godtliebsen, F. 2020. Reinforcement learning application in diabetes blood glucose control: A systematic review. *Artificial intelligence in medicine*, 104: 101836.

Wang, Z.; Schaul, T.; Hessel, M.; Hasselt, H.; Lanctot, M.; and Freitas, N. 2016. Dueling network architectures for deep reinforcement learning. In *International conference on machine learning*, 1995–2003. PMLR.

Weng, J.; Chen, H.; Yan, D.; You, K.; Duburcq, A.; Zhang, M.; Su, H.; and Zhu, J. 2021. Tianshou: a Highly Modularized Deep Reinforcement Learning Library. *arXiv preprint arXiv:2107.14171*.

7 Appendix

Related Work on Offline Reinforcement Learning

Offline RL (i.e. batch-RL or fully off-policy RL) learns from a pre-collected dataset without interaction with an environment. Offline RL literature tends to solve the out-of-distribution(OOD) challenge [Fujimoto, Meger, and Precup 2019]. Conservative Q learning [Kumar et al. 2020] learns a lower bound of the actual Q function and proves that improving such a lower bound can effectively improve its policy. BCQ [Fujimoto et al. 2019] considers a Q learning with policy constraints. Instead of directly applying the action with the highest Q value, BCQ leans towards choosing a 'safe' action presented in the dataset. Kumar et al. [2019] proposed Bootstrapping Error Accumulation Reduction (BEAR) and pointed out that BCQ's constraint might limit the policy from improving when the offline behaviour is not near-optimal. BEAR loosen the constraint of BCQ by placing non-zero probability mass on actions with non-negligible behaviour policy density. Apart from the approaches mentioned above, which reduce OOD by state-action constraints, IQL [Kostrikov, Nair, and Levine 2021] prevents the agent from using OOD Q estimation to update and therefore avoid the OOD challenge to some extent. For a reason that game benchmark datasets (e.g., [Fu et al. 2020]) are collected mainly by RL agents while exploring online, they do not suffer from class imbalance or exploration issues like in DTR. Thus, whether they work well on DTR datasets is still being determined.

Offline RL-based DTR in healthcare is a new domain with limited literature. Guez et al. [2008] studied epilepsy treatment by implementing standard Fitted-Q iteration (FQI). Liu et al. [2017] applied offline DQN using medical registry data to treat graft-versus-host disease. Fatemi et al. [2021] introduces the concept of dead-end discovery (DeD) to identify ineffective treatment strategies, where "dead-end" means the patient will die regardless of any future treatments. Using a binary outcome as the reward, they implement Q learning to predict the dead-ends along the treatment trajectory. DeD can only inform the risk of action and does not recommend the optimal action as in our proposed work. DeD also cannot differentiate actions if they are classified as non-dead-ends. Sonabend et al. [2020] proposed a Bayesian RL framework leveraging uncertainty quantification for offline policy learning and provided interpretability to the approximated posterior distribution. However, the choice of prior distribution significantly affects model performance.

Although the above-related works provide offline RL in DTR, they do not explicitly address the challenges discussed in Section 1. Our proposed framework, *NurSpecialist*, transforms the single-agent DTR problem into a fully cooperative multi-agent setup and develops a novel communication mechanism to eliminate miscommunication between agents. The involvement of the *nurse* agent in our framework helps reduce unwanted treatment changes. We further prove that the *nurse* and *specialist* agents can be updated separately by any existing single-agent algorithms without violating the optimal Bellman equation for independent agent learning.

Assumptions and Proofs

Assumption on Human Doctor's Policy we assume the human doctor is always exploiting its policy to save patients, although a human doctor's policy might be non-optimal. Under this assumption, we have

$$\sum_{j\in\mathcal{A}} \beta(\boldsymbol{s}_t, a_j) Q_{sp}(\boldsymbol{s}_t, a_j) = \max_{a\in\mathcal{A}} Q_{sp}(\boldsymbol{s}_t, a). \tag{8}$$

since

$$\beta_{sp} = \begin{cases} 1 \text{ if } \underset{a\in\mathcal{A}_{sp}}{argmax}\, Q_{sp}(s,a), \\ 0 \text{ otherwise.} \end{cases} \tag{9}$$

Definition of a *nurse* MDP The value function under human doctor's policy for the *specialist* MDP is

$$\begin{aligned} V^{\beta_{sp}}(\boldsymbol{s}_t) &= \sum_{a\in\mathcal{A}} \beta^{sp}(\boldsymbol{s}_t, a) Q_{sp}(\boldsymbol{s}_t, a) \\ &= \beta_{sp}(\boldsymbol{s}_t, a_i) Q_{sp}(\boldsymbol{s}_t, a_m) \\ &+ \sum_{a_j\in\mathcal{A}\setminus a_m} \beta_{sp}(\boldsymbol{s}_t, a_j) Q_{sp}(\boldsymbol{s}_t, a_j), \\ &\text{for } \forall a_{t-1} = a_m \end{aligned} \tag{10}$$

where a_{t-1} is the action taken in the previous step. Therefore we can define an induced *nurse* value function V^{β_n}, policy π^{β_n} and function Q_n as

$$\begin{aligned} &V^{\beta_n}(\boldsymbol{s}_t | a_{t-1} = a_m) \\ &= \sum_{k\in A_n} \beta_n(\boldsymbol{s}_t, k | a_{t-1} = a_m) Q_n(\boldsymbol{s}_t, k | a_{t-1} = a_m). \end{aligned} \tag{11}$$

By comparing Eq 10 and Eq 12, for $\forall a_{t-1} = a_m$, we define

$$\beta_n(\boldsymbol{s}_t, k_0 | a_{t-1} = a_m) \doteq \beta_{sp}(\boldsymbol{s}_t, a_m). \tag{12}$$

$$Q_n(\boldsymbol{s}_t, k_0 | a_{t-1} = a_m) \doteq Q_{sp}(\boldsymbol{s}_t, a_m) \tag{13}$$

$$\begin{aligned} &\beta_n(\boldsymbol{s}_t, k_1 | a_{t-1} = a_m) Q_n(\boldsymbol{s}_t, k_1 | a_{t-1} = a_m) \\ &\doteq \sum_{j\in\mathcal{A}\setminus m} \beta_{sp}(\boldsymbol{s}_t, a_j) Q_{sp}(\boldsymbol{s}_t, a_j) \end{aligned} \tag{14}$$

such that $V^{\beta_n}(\boldsymbol{s}_t | a_{t-1} = a_m) = V^{\beta_{sp}}(\boldsymbol{s}_t)$. It is to say, when the *nurse* decides not to change treatment, its Q value is equivalent to the original Q value on taking its previous action; on the other hand, when the *nurse* decides to change treatment, its Q value is equivalent to using the highest Q value among actions that are different from its previous action.

Lemma 1. In discrete action space, $\mathcal{A}_{sp}$, *nurse* actions from an induced *nurse* policy can be collected by comparing the current action with its previous action.

Proof Since we have assumed the human doctor's policy is fully exploitation, the induced *nurse* policy is fully exploitation as well. Substituting Eq 9 into Eq 12, the two components of *nurse* policy are

$$\begin{aligned} \beta n(s,k_0) &= \begin{cases} 1 \text{ if } \underset{a\in\mathcal{A}_{sp}}{argmax}\, Q_{sp}(s,a) = a_{t-1}, \\ 0 \text{ otherwise.} \end{cases} \\ \beta n(s,k_1) &= 1 - \beta n(s,k_0) \end{aligned} \tag{15}$$

Hence, sampling from $\beta n(s,k)$ is equivalent to comparing the current action with its previous action.

Lemma 2. An optimal *nurse* policy π^{n*} is equivalent to an optimal *specialist* policy π^{sp*} when $p'(\beta^*_{sp})$.

Proof. The optimal policy for a *nurse* agent is

$$\pi^{n*} = \begin{cases} 1 \text{ if } \underset{k\in\mathcal{A}_n}{argmax}\, Q^*_n(s,k), \\ 0 \text{ otherwise.} \end{cases} \tag{16}$$

We prove Lemma 2 by proving $argmax_{k\in A_n} Q^*_n(s,k) = argmax_{a\in A_{sp}} Q^*_{sp}(s,a)$. Since an optimal *specialist* agent induces the *nurse* agent, it holds that for $\forall \upsilon^*_{t-1} \in A_{sp}$,

$$\begin{cases} Q^*_n(s,k_0|\upsilon_{t-1}=\upsilon^*_{t-1}) = Q^*_{sp}(s,\upsilon^*_{t-1}) \\ Q^*_n(s,k_1) = \underset{a\in A_{sp}}{argmax}\, Q^*_{sp}(s,a) \\ Q^*_{sp}(s,\upsilon^*_{t-1}) = \underset{a\in A_{sp}}{argmax}\, Q^*_{sp}(s,a) \\ \text{if } Q^*_n(s,k_0|\upsilon_{t-1}=\upsilon^*_{t-1}) > Q^*_n(s,k_1|\upsilon_{t-1}=\upsilon^*_{t-1}) \end{cases} \tag{17}$$

Thus we can derive the equality

$$\begin{aligned} &\underset{k\in A_n}{argmax}\, Q^*_n(s,k) \\ &= \underset{k\in\mathcal{A}_n}{argmax} \left[Q^*_n\left(s,k_0|\upsilon_{t-1}=\upsilon^*_{t-1}\right), Q^*_n(s,k_1)\right] \\ &= \underset{k\in\mathcal{A}_n}{argmax} \left[Q^*_{sp}\left(s,\upsilon^*_{t-1}|\upsilon_{t-1}=\upsilon^*_{t-1}\right), \underset{a\in A_{sp}}{argmax}\, Q^*_{sp}(s,a)\right] \\ &= \underset{a\in A_{sp}}{argmax}\, Q^*_{sp}(s,a). \end{aligned} \tag{18}$$

Remarks on Vasopressor-Intravenous dosage regime for sepsis treatment 35 variables are extracted as the observation space, including demographics, vital signs, laboratory tests and fluid balance. Patients' observations were coded as multidimensional discrete-time series with 2-hour time steps. Two interventions, i.e., mechanical ventilation and vasopressor, are taken as the action space. Each intervention is binarised, representing the "ON/OFF" status of applying the intervention in a specific hourly interval. Usage of ventilation and vasopressor on patients are correlated and with no order. Therefore we re-group the actions space into 4 classes, i.e., {*None*, *Vent*, *Vaso*, *Vent+Vaso*}.

Mean is taken if the observational feature contains replicated samples within an hour. Missing data is imputed in an in-admission forward fill-in manner. The global mean is applied if a feature does not appear in a patient's admission. Observation data is scaled in the range of [0, 1] by *minmax* normalisation.

Remarks on Data Curation and Preprocessing

The oxygen therapy cohort natively contains 3 subsets of data, where set A and set B are used for training and validation, respectively and set C for testing. We combine set A and set B to form a larger training-validation set for k-fold validation and remain set C as the testing set. A detailed data description can be found in table 3.

The state space of this task contains 34 features, including demographics, vital signs and those that are related to the ventilation weaning, such as SaO_2 (O_2 saturation in haemoglobin (%)) and PaO_2 (Partial pressure of arterial O_2 (mmHg)). The action space is binary, i.e., whether to wean mechanical ventilation. Overuse/underuse/misuse of mechanical ventilation will likely induce harm to patients [Esteban et al. 2013][Kester and Stoller 1992], which will reflect patient mortality. Therefore, we use patient survival as the reward to be optimised. In practice, the reward function is a function of the remaining admission length of stay after the first 48 hours:

$$r_i(t) \coloneqq \begin{cases} 100\gamma^{T_i-2} & \text{if } t = T_i \text{ and the patient survived} \\ -100\gamma^{T_i-2} & \text{if } t = T_i \text{ and the patient deceased} \\ 0 & 0 \le t < T \end{cases}, \tag{19}$$

where i is the i^{th} patient, t is the time step, and T_i is the total admission time in days of patient i. Intuitively, the agent is encouraged to minimise the length of stay if the patient can be saved and maximise the survival time for deceased patients.

For the Vasopressor-Intravenous Dosage Regime cohort, We develop a systematic way to split the training set, validation set and testing set for fairly evaluating policies on imbalanced data.

1. **Stratification**. Patients are stratified according to age, gender, phenotype, and admission outcome. Each criterion stratifies patients into binarised groups, and by permuting the criteria, we can divide the whole patient cohort into 16 groups.
2. **Split**. We select 20% of patients as the testing set. The rest 80% of patients are further divided into 5 equal sets for 5-fold cross-validation. Patients from each stratified group are taken out and then combined as a sub-set. If the number of patients is divided with a remanent, the remaining patients will be put into the testing set. In this way, we derive a training, validation and testing set that almost has an equal proportion of stratified patient types, such that the distributions of patients are even.

Remarks on Code and Dataset Availability

* To access the dataset, the user will need to be credentialed for MIMIC-III access through physionet. Instructions for that are available on the official website https://physionet.

Table 2: Data Description of the Oxygen Therapy in the General ICU Cohort

	Set A	**Set B**	**Set C**
Age, years(Mean/Std)	64.66 (17.65)	65.20 (16.90)	65.11 (16.96)
Male Gender	54.14	56.61	56.50%
Length of Stay, days (Mean/Std)	10.92(6.70)	10.81(6.67)	10.59(6.54)
In-hospital Mortality	13.91%	14.16%	14.94%

Table 3: Data Description of the Vasopressor-Intravenous Dosage Regime for Sepsis Treatment Cohort

Age, years(Mean/Std)	64.47 (16.73)
Male Gender	56.62 %
Weight, Kg (Mean/Std)	83.82 (25.17)
Readmission	30.90 %
In-hospital mortality	17.40%
90 days mortality	27.23%

Table 4: Stratification

age	> 60	$<= 60$
gender	Female	Male
readmission	Yes	No
outcome	Survived	deceased

org/content/mimiciii/1.4/. Our data curation and preprocessing code rely on *Predicting Mortality of ICU Patients: The PhysioNet/Computing in Cardiology Challenge 2012*, which can be found at https://physionet.org/content/challenge-2012/1.0.0/ for the mechanical ventilation cohort and https://github.com/uribyul/py_ai_clinician for the Vasopressor-Intravenous dosage regime for sepsis treatment cohort. Please note that the link above might not be permanent, depending on the version update of open-source repositories.

Network Realization

Observation O_t consists of two segments: dynamic features o_{dt} and static featureso_{st}. In a DTR problem, dynamic features refer to features that change over time, such as vital signs and blood tests, and static features refer to those with no change or ignorable changes during the admission, such as age, readmission status and gender.

A backbone model f_b processes the two types of features independently. We use Recurrent Neural Network(RNN) to extract dynamic information and Multi-Perceptron (MLP) to extract static information. Then, the two extracted representations are concatenated together and inserted into another set of MLP layers to produce a mixed-modality state representation.

Non-communicative policy models (both baseline models and *NurSpecialist* models) are parameterised by 3 layers of MLP with ReLu activation, communicative *NurSpecialist* uses 2 layers of LSTM connected to 3 layers of MLP for *nurse*-specialist communication.

Realisation of Behavioural Cloning Agent

Behavioural cloning tends to map directly from observation to a human doctor's action instead of learning the value/policy for decision-making.

Regarding the oxygen therapy task, the behavioural cloning model uses the same backbone model structure (trained from scratch) to extract state representation. A classifier replaces the policy layer with Log-Softmax activation. We train the model by minimising the negative log-likelihood(NLL) loss between predicted actions and human doctors' actions.

The behavioural agent for the sepsis treatment task is reproduced by empirical occurrence counting presented in [Komorowski et al. 2018]. Firstly, continuous observations were discretised into 400 states by K-Mean clustering. The number of clusters is chosen based on the Bayesian information criterion (BIC).

Model Training, Validation and Model Selection

All models are trained based on Pytorch [Paszke et al. 2017], and Tianshou [Weng et al. 2021] with Adam optimiser. The choice of standard model hyperparameters can be found in table 5. In the table, seed refers to both NumPy [Harris et al. 2020] seed and PyTorch seed. The observation window is the window length T for $\mathtt{H}_T$ (See section 4). The hyperparameter choice is not listed exhaustively since each stage of training/testing has its independent hyperparameter set.

Table 5: Hyperparameter Selection

Table 6: Hyperparameter choice for model training.

Adams optimizer	$\beta_1 = 0.9, \beta_2 = 0.999$ $\epsilon = 10^{-8}$
Seed	[1, 500, 5000, 12345, 67890]
Learning rate	[0.001, 0.005, 0.0001]
Batch size	[128, 256, 512]
Observation window	[1, 12, 24]

For training on each set of hyperparameter, 5-fold cross-validation is conducted on patient-stratified data split(See section 7). We set an early stop to avoid overfitting. Suppose a selected criterion on the validation set stops improving for 20 epochs consecutively. In that case, the training process will be terminated, and the model in which the epoch achieves the best criterion performance is selected for testing. For single-agent models, the best criterion means the lowest TD error. For *NurSpecialist* models, we select the lowest TD

error of the *specialist* agent as the best criterion. Note that we do not take F1-score as a criterion for model selection because the initiative of reinforcement learning is not to imitate human doctors' behaviour. In the testing part, metrics are computed by taking the best 10 models with each method with 5 different seeds, respectively.

All experimental work is conducted on a workstation with CPU intel i9-9900k, GPU Nvidia RTX 2080Ti, and 32GB of RAM. Each epoch of model training takes approximately 10-50 seconds. Training speed may change with respect to hardware configuration.

Importance Sampling for Off-policy Evaluation In our case, we want to estimate the average reward a learnt policy $\pi(s, a)$ can achieve by using samples (i.e., $(\boldsymbol{o}_t^i, a_t^i, r_t^i) \sim \mathcal{D}$) collected under human doctor's treatment policy β. Since we do not have access to the distribution of human doctor's policy, a learnt behavioural policy $\hat{\beta} \rightarrow \beta$ is replaced. Training details and model selection can be found in 7. The WIS estimator is of the form

$$\hat{v}_{WIS}^{\pi} = \frac{1}{\sum_{i=1}^{N} w_i} \sum_{i=1}^{N} \left(w_i \sum_{t=1}^{T_n} \gamma^t r_t^i \right), \tag{20}$$

where $w_i = \prod_{t=1}^{T_n} \frac{\pi(s_t^i, a_t^i)}{\hat{\beta}(s_t^i, a_t^i)}$ is the ratio multiplication for patient i. $\hat{v}_{WIS}^{\pi}$ represents the future expected return for a target policy π, higher the better. Empirically, we found that the per-step ratio $\frac{\pi(s_t^i, a_t^i)}{\hat{\beta}(s_t^i, a_t^i)}$ is extremely high on only a small group of patients (approximately 40 over 2,000 patients), which will result in huge variance to the overall estimated value $\hat{v}_{WIS}^{\pi}$. Therefore, we exclude patients whose WIS ratio w_i is higher than 10^6.

For value-based learning algorithms, the model does not explicitly produce a policy distribution but gives a set of Q values. We empirically use the following equation to generate a policy from the corresponding set of Q values:

$$\pi_{SA}(\boldsymbol{s}, a) = \sigma(\boldsymbol{Q}), \tag{21}$$

, where $\sigma(x)_m = \frac{\exp x_m}{\sum_{m'=1}^{M} x_{m'}}$ is the softmax function, and $\boldsymbol{Q} = [Q(\cdot, a), a \in \mathbb{A}]^T$ is the Q vector including all actions at the current state.

Similarly, the *NurSpecialist* policy is derived by applying Eq 21 to both *nurse* and *specialist* agent and take a sum weighted by the *nurse* policy:

$$\pi_{NS}(\boldsymbol{s}, a_m) = \delta(a_{t-1} = a_m)\pi_n(\boldsymbol{s}, k_0) + \pi_n(\boldsymbol{s}, k_1)\pi_{sp}(\boldsymbol{s}, a_m), \tag{22}$$

where

$$\pi_n(\boldsymbol{s}, k) = \sigma(\boldsymbol{Q_n}), \tag{23}$$

$$\pi_{sp}(\boldsymbol{s}, a_m) = \sigma(\boldsymbol{Q_{sp}}), \tag{24}$$