

Berdasarkan sumber-sumber yang Anda berikan, berikut adalah referensi yang paling relevan untuk menentukan **Threshold**, **Reward/Penalty**, **Action**, dan **State** dalam Reinforcement Learning, dikhususkan untuk konteks jaringan (SDN/Network Slicing):

1. Penentuan Reward Function & Penalty (SLA-Aware)

Bagian ini paling krusial untuk memastikan model RL Anda (PPO/DQN) mematuhi prioritas Port 4 (Healthcare) dan Port 2 (Camera).

- **Jain et al. 1 ("Safety in DRL-Based Congestion Control..."):**
 - **Relevansi:** Sumber ini sangat detail membahas desain *reward function* linear yang menyeimbangkan *throughput*, *latency*, dan *packet loss*.
 - **Ide untuk Anda:** Mereka menggunakan bobot (ω) untuk mengatur penalti. Anda bisa mengadaptasi rumus mereka: $R = \omega_1 \cdot \text{Throughput} - \omega_2 \cdot \text{Latency} - \omega_3 \cdot \text{Loss}$. Untuk Port 4 (Healthcare), berikan bobot ω_2 (latency) yang sangat besar.
 - **Threshold:** Mereka menggunakan ambang batas (threshold) berbasis RTT historis untuk menentukan "unsafe state" dan memberikan penalti besar jika dilanggar.
- **Zeng 2 ("Near-Real-Time Resource Slicing..."):**
 - **Relevansi:** Fokus pada *regret-based optimization*.
 - **Ide untuk Anda:** Menggunakan kebalikan dari *regret* sebagai *reward*. Jika *latency* melebihi batas SLA, *regret* tinggi, sehingga *reward* rendah. Ini cocok untuk skenario *network slicing* Anda di mana setiap slice punya kebutuhan QoS berbeda.
- **Raftopoulos et al. 3 ("DRL-based Latency-Aware Network Slicing..."):**
 - **Relevansi:** Secara spesifik membahas *latency-aware slicing*.
 - **Ide untuk Anda:** Mereka menggunakan fungsi *Sigmoid* dalam *reward* untuk memberikan penalti "lambut" saat mendekati batas SLA dan penalti keras saat melanggar. Ini membantu agen belajar lebih stabil daripada sekadar penalti biner (0/1).
- **Chen et al. 4 ("Slicing Resource Allocation Based on Dueling DQN..."):**
 - **Relevansi:** Menggunakan **Reward Clipping**.
 - **Ide untuk Anda:** Agar *gradient* tidak meledak saat *reward* terlalu besar atau terlalu kecil (misal saat *throughput* sangat tinggi), gunakan mekanisme *clipping* (memotong nilai reward ke rentang tertentu, misal -1, 1).

2. Definisi State Space (Fitur Jaringan)

Untuk dataset 37 fitur Anda, referensi ini membantu memilih mana yang relevan sebagai input *State*.

- **Iqbal et al. 5 ("A Scalable Reinforcement Learning Framework..."):**
 - **Relevansi:** Membahas *State Space* untuk *Healthcare IoT*.
 - **Ide untuk Anda:** Mereka menyarankan menyertakan **Buffer Occupancy (Queue Size)** dan **Recent SINR/Channel Quality** ke dalam *state*. Dalam dataset Anda, ini setara dengan fitur *util_rx_pct* (utilisasi/antrian) dan *drop* (indikator kualitas kanal).
- **Abdulazeez et al. 6 ("Offloading Mechanisms Based on RL..."):**
 - **Relevansi:** Memberikan tinjauan komprehensif tentang bagaimana berbagai paper mendefinisikan *state* untuk *offloading* dan alokasi resource.
 - **Ide untuk Anda:** Gunakan referensi ini untuk menjustifikasi mengapa Anda memilih fitur tertentu (seperti *rx_mbps*, *delay*, *drop*) sebagai representasi kondisi jaringan.

3. Definisi Action Space & Threshold

Bagaimana agen mengubah parameter jaringan (SDN Action).

- **Luo et al. 7 ("Reinforcement Learning for Traffic Signal Control in Hybrid Action Space"):**
- **Relevansi:** Membahas ruang aksi *Hybrid* (Diskrit dan Kontinu).
- **Ide untuk Anda:** PPO Anda bisa menggunakan *Continuous Action* (menentukan *policing_rate* secara presisi, misal 5.2 Mbps), sedangkan DQN/Dueling DQN Anda harus menggunakan *Discrete Action* (misal: naikkan 10%, turunkan 10%). Referensi ini bagus untuk menjelaskan perbedaan perlakuan aksi antar model di paper Anda.
- **Xia et al. 8 ("Genet: Automatic Curriculum Generation..."):**
- **Relevansi:** Menentukan *Threshold* kesulitan lingkungan.
- **Ide untuk Anda:** Alih-alih menetapkan *threshold* statis untuk semua kondisi, gunakan pendekatan *baseline* aturan (rule-based). Jika performa agen RL lebih buruk dari aturan statis sederhana (misal FIFO), berikan penalti tambahan.

Rekomendasi Spesifik untuk Paper Anda:

1. **Untuk Menentukan Threshold SLA (Reward/Penalty):** Gunakan pendekatan **Jain et al. 1** dan **Raftopoulos et al. 3**. Gabungkan konsep *weighted linear reward* (untuk throughput Port 2) dengan *sigmoid penalty* (untuk latency Port 4).
2. **Untuk Desain State:** Kutip **Iqbal et al. 5** untuk menjustifikasi penggunaan fitur antrian (*util_rx_pct*) dan delay (*delay_ms*) sebagai input utama agen RL untuk kasus IoT Kesehatan.
3. **Untuk Normalisasi State:** Gunakan referensi **Huang et al. 9** yang menjelaskan pentingnya normalisasi matriks trafik (seperti Min-Max scaling) agar DQN/PPO lebih cepat konvergen.