

Forecast-Driven Doctor Scheduling in Primary Healthcare: Ensemble Learning, a Reinforcement Learning Agent, and an Exact Baseline

M. Agus Munandar¹⁾, Terttiaavini^{2*)}

^{1,2)}Universitas Indo Global Mandiri, Indonesia

¹⁾agus@uigm.ac.id, ²⁾avini.saputra@uigm.ac.id

ABSTRACT

Doctor scheduling in primary healthcare facilities is challenging because patient visit volume fluctuates due to seasonality, holidays, weather, and unplanned staff absences, while rule-based scheduling used in many healthcare systems remains static and unresponsive. This study builds and rigorously validates a forecast-driven scheduling pipeline for a primary healthcare center (Puskesmas Talang Ratu, Palembang, Indonesia), using five years (2019–2024) of visit, attendance, weather, and holiday records engineered into 46 temporal, operational, and contextual features. Six ensemble regressors were compared using a chronological split; Random Forest generalised best (MAE = 7.08, RMSE = 10.44, R^2 = 0.82, next-day visits). The forecast was embedded in a Deep Q-Network (DQN) scheduling agent and benchmarked, under a leak-free walk-forward forecasting procedure with the forecast-to-decision day alignment corrected, against three baselines: the facility's existing rule-based practice, a constraint-aware forecast-to-staffing rule, and a myopic cost-optimal integer program that solves the DQN's own objective exactly each day. Over a 301-day held-out period, the exact optimizer achieved the best capacity match (MAE 8.11) and the lowest objective cost, ahead of the constraint-aware rule (MAE 8.42) and the DQN policy (MAE 9.90–10.26 across reward-scaling and seed variants), all of which greatly outperformed the existing rule-based practice (MAE 20.61). The DQN did not surpass the exact optimizer even on its own training objective, a result that held across four random seeds, extended training, and reward rescaling. We trace this to a structural property of the environment: because staffing decisions do not influence future states or rewards, the task reduces to a per-day (contextual-bandit-like) optimization rather than a genuinely sequential one, removing the advantage reinforcement learning derives from learning delayed returns. The contribution is therefore a leak-free, day-aligned forecasting-to-scheduling pipeline and an empirically grounded account of when reinforcement learning is, and is not, warranted for demand-responsive capacity allocation in primary healthcare.

Keywords: Deep Q-Network; Doctor Scheduling; Ensemble Learning; Healthcare Systems; Information Leakage; Patient Visit Prediction

INTRODUCTION

Primary healthcare facilities such as Puskesmas play a central role in Indonesia's healthcare system, yet their doctor scheduling practices are frequently managed manually using fixed or rule-based approaches (Fatikasari et al., 2024; Hendiawati et al., 2023; Pratiwi & Siambaton, 2022). This static approach struggles to accommodate fluctuations in daily patient visits caused by seasonal patterns, national holidays, weather conditions, and sudden staff absences. When demand exceeds scheduled capacity, patient waiting times increase and service quality declines; when demand falls short, medical staff are underutilized and operational costs rise unnecessarily (Wang & Demeulemeester, 2023).

Artificial intelligence has been widely applied to healthcare scheduling problems, ranging from classical optimization techniques such as genetic algorithms (Priatna et al., 2023) and multi-objective evolutionary methods (Mahariani, 2023) to more recent data-driven approaches based on machine learning and

* Corresponding author



[Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.](https://creativecommons.org/licenses/by-nc-sa/4.0/)

reinforcement learning (Yu et al., 2023). Reinforcement learning in particular has been applied to emergency department patient scheduling (Lee & Lee, 2020), ambulance dispatch (Liu et al., 2020), cloud-based healthcare resource scheduling (Li et al., 2022), outpatient scheduling using multi-agent deep reinforcement learning (Zuo et al., 2024), and patient scheduling combined with combinatorial optimization in hospital settings (Liu et al., 2025). Most such studies benchmark the learned policy against static rules only; few compare against an exact solution of the same objective, so it is rarely established whether the learning component itself, rather than the underlying forecast, is responsible for reported gains.

This study addresses that gap directly for daily general-practitioner scheduling at a Puskesmas. Its contribution is fivefold: (1) a comparative evaluation of six ensemble learning algorithms for forecasting daily patient visits using five years of operational data; (2) a Deep Q-Network scheduling agent with an explicitly specified state, action, transition and reward structure; (3) a leak-free, day-aligned evaluation protocol — out-of-sample walk-forward forecasting and forecast-to-decision-day alignment — developed after we identified that a naive implementation of both introduces substantial, easily overlooked bias; (4) a benchmark against a constraint-aware forecast-to-staffing rule and a myopic cost-optimal integer program that solves the agent's own objective exactly, used to isolate the specific contribution of learning; and (5) a structural diagnosis, supported by seed-repeated experiments, of why reinforcement learning does not outperform exact optimization in this problem once the objective is a per-day capacity match with no cross-day coupling. We report this last point because it was the actual outcome of a rigorous evaluation, not the outcome we set out to demonstrate, and because it directly answers the question of whether a learned policy provides value beyond a direct forecast-to-staffing rule — a question this study is, to our knowledge, one of the few in this literature to test explicitly rather than assume.

LITERATURE REVIEW

Ensemble learning combines multiple base learners to improve predictive accuracy and generalization compared to a single model. Bagging-based methods such as Random Forest reduce variance by aggregating predictions from multiple decision trees trained on bootstrapped samples, while boosting-based methods such as Gradient Boosting (Friedman, 2001), XGBoost, and LightGBM (Ke et al., 2017) sequentially correct the errors of previous learners. Stacking further combines heterogeneous base models through a meta-learner. Systematic reviews report that ensemble methods generally outperform single models in healthcare-related prediction tasks, including disease prediction (Mahajan et al., 2023) and electronic health record screening (Stevens et al., 2023), owing to their robustness against noise and their ability to capture non-linear relationships in operational data.

Reinforcement learning (RL) has increasingly been applied to healthcare scheduling and resource allocation problems because it allows an agent to learn adaptive policies through interaction with an environment rather than relying on fixed rules (Yu et al., 2023). Deep Q-Network (DQN), introduced by Mnih et al. (2015), extends Q-learning with deep neural function approximation and has since been applied to dynamic scheduling problems in manufacturing (Liu et al., 2022; Fu et al., 2024) and healthcare settings, including emergency department patient scheduling (Lee & Lee, 2020), ambulance dispatch (Liu et al., 2020), cloud-based healthcare resource scheduling (Li et al., 2022), and outpatient scheduling using multi-agent deep reinforcement learning (Zuo et al., 2024).

A basic premise of reinforcement learning is that current actions influence future states or rewards, which is what justifies learning value functions over immediate greedy alternatives (Sutton & Barto, 2018). Several deployed formulations of scheduling-as-RL, including the two-agent outpatient scheduler of Zuo et al. (2024) and the combinatorial patient scheduler of Liu et al. (2025), retain genuine cross-period coupling (queue carry-over, resource contention across time steps). Whether a daily doctor-count decision retains this coupling, or reduces to a sequence of independent per-day decisions once framed around a next-

* Corresponding author



[Creative Commons Attribution-NonCommercial-ShareAlike 4.0
International License.](https://creativecommons.org/licenses/by-nc-sa/4.0/)

day demand forecast, is an empirical question specific to the environment design; this study tests it directly rather than assuming an RL formulation is warranted.

Several studies have specifically examined doctor and medical staff scheduling. Genetic-algorithm-based approaches have been used to automate shift scheduling (Priatna et al., 2023), while NSGA-II has been applied to multi-objective hospital operating room scheduling (Mahariani, 2023). Wang and Demeulemeester (2023), in a literature review of simulation optimization in healthcare resource planning, found that emergency care is the most frequently studied domain and that discrete-event simulation combined with genetic-algorithm-based search is the most common methodological pairing, though reported outcomes vary substantially across studies and settings. Locally, prior work has applied greedy algorithms (Pratiwi & Siambaton, 2022) and web-based scheduling systems (Hendiawati et al., 2023; Fatikasari et al., 2024) to hospital and Puskesmas contexts, and a reinforcement-learning hyper-heuristic approach has been explored for staff scheduling at a maternal and child hospital (Karami, 2018).

Despite this growing body of work, most existing studies either focus on staff categories other than general-practitioner doctors, apply optimization or reinforcement-learning methods without an integrated, leak-free demand-forecasting component, or benchmark reinforcement learning exclusively against static rules, a weak comparison that cannot separate the value of the forecast from the value of the learning algorithm. This study addresses that gap by combining ensemble-based demand forecasting with a DQN scheduling agent, evaluated under a leak-free, day-aligned protocol and benchmarked against an exact optimizer, for daily general-practitioner scheduling at a Puskesmas using five years of real operational data.

METHOD

Research Design and Data Source

This study employed a quantitative, design-science research approach consisting of two integrated stages: (1) patient visit forecasting using ensemble learning, and (2) adaptive scheduling optimization using reinforcement learning, benchmarked against non-learning baselines. The research object was Puskesmas Talang Ratu, Palembang, Indonesia. Four datasets covering the period 2019–2024 were used: daily patient visit records, doctor attendance records, daily weather data, and a national/collective-leave holiday calendar. All experiments were conducted in Python with scikit-learn, XGBoost, and LightGBM for the ensemble stage and Gymnasium with Stable-Baselines3 for the reinforcement learning stage.

Preprocessing and Feature Engineering

Raw datasets were merged on the date field. Missing weather values were linearly interpolated and forward/backward filled; missing visit, attendance and holiday-flag values were treated as zero (no recorded activity). Closed days (Sundays and national holidays) had visit counts fixed at zero by construction rather than imputed, since the facility does not operate on those days. No visit-count outliers were removed — extreme days (e.g. post-holiday surges) are genuine operational events the forecaster must learn, not measurement error. Temporal features (day of week, Sunday indicator, non-operational day indicator, national holiday and collective-leave indicators) were derived from the calendar attributes. Operational and contextual features included the number of available doctors, the number of absent doctors, and scaled weather variables (temperature, humidity, precipitation, wind speed). Demand-signal features were constructed from the historical visit series, comprising lagged visit counts (lag-1, lag-2, lag-3, lag-7) and rolling means (7-day and 14-day windows) to capture short-term memory and trend. All numeric features were standardized using a scaler fitted exclusively on the training partition to avoid data leakage. The prediction target was next-day visit count (`jumlah_kunjungan_total` shifted by -1 day), not same-day count; this choice is operationally natural (staffing is planned a day ahead) but, as Section “Forecast-to-Decision-Day Alignment” details, requires care in how the forecast is subsequently consumed by any downstream

* Corresponding author



[Creative Commons Attribution-NonCommercial-ShareAlike 4.0
International License.](https://creativecommons.org/licenses/by-nc-sa/4.0/)

decision system. After feature construction and removal of incomplete rows introduced by lagging, the final dataset comprised $N = 2,003$ daily observations spanning 8 July 2019 to 30 December 2024, with 46 input features (the complete list is provided in the Appendix). The dataset was split chronologically into training ($n = 1,402$; 70%; 8 Jul 2019–9 May 2023), validation ($n = 300$; 15%; 10 May 2023–4 Mar 2024), and testing ($n = 301$; 15%; 5 Mar 2024–30 Dec 2024) partitions.

Ensemble Learning for Patient Visit Prediction

Six ensemble regression algorithms were trained and compared: Random Forest (bagging), Gradient Boosting and AdaBoost (sequential boosting), XGBoost and LightGBM (gradient boosting frameworks optimized for speed and regularization), and a Stacking ensemble combining all five base learners through a Ridge meta-regressor. Hyperparameters were set to widely used default-range configurations rather than exhaustively tuned via grid or random search, as summarized in Table 1; a fixed random seed (42) was used for all stochastic components (data splitting, bootstrap sampling, and model initialization) to ensure reproducibility. Model performance was evaluated using Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and the coefficient of determination (R^2) on both the training and testing partitions.

Table 1. Hyperparameter configuration of the six ensemble models

Model	Key Hyperparameters
Random Forest	$n_estimators=300$, $max_depth=10$, $random_state=42$
Gradient Boosting	$n_estimators=300$, $learning_rate=0.05$
AdaBoost	$n_estimators=200$, $learning_rate=0.1$
XGBoost	$n_estimators=300$, $learning_rate=0.05$, $max_depth=6$
LightGBM	$n_estimators=300$, $learning_rate=0.05$
Stacking	Base learners above; meta-learner = $RidgeCV(\alpha \in \{0.01, 0.1, 1, 10\})$

The Stacking meta-learner was fit on out-of-fold base-model predictions generated with TimeSeriesSplit (3 splits) rather than ordinary K-fold cross-validation, so that no fold's training data follows its own test data chronologically; the first split's training block, by construction, contributes no out-of-fold prediction and was excluded from the meta-learner's fit. This corrects an inconsistency in an earlier version of this manuscript, which reported a 5-fold cross-validation in text against a $cv=3$ setting in code; both have now been unified as the single time-aware 3-split procedure described above, and Table 1 reflects the corrected setting.

Forecast-to-Decision-Day Alignment

Because the forecasting target is next-day visits, a model trained on (features at day i , target = visits at day $i+1$) produces, when queried at day i , a forecast for day $i+1$, not day i . An earlier version of this pipeline compared that forecast directly against day i 's actual visits when computing scheduling rewards and evaluation metrics, a one-day misalignment. We detected it by observing that, once the forecaster was evaluated leak-free (see below), the correlation between forecast and same-day actual visits fell to near zero

* Corresponding author



[Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.](https://creativecommons.org/licenses/by-nc-sa/4.0/)

($r = -0.02$) despite the forecaster's genuine next-day test-set skill ($r = 0.91$, matching the $R^2 = 0.82$ in Table 2); a leaky, full-data-refit forecaster had masked the misalignment by overfitting to whichever day's value it was compared against. The fix does not require retraining the forecaster: every decision epoch i now consumes the observation available at day i to choose staffing for day $i+1$, hard constraints (holiday/Sunday/collective-leave) are evaluated against day $i+1$'s calendar, and every reported metric compares the resulting capacity against ACTUAL at day $i+1$. All results in this manuscript use this corrected alignment.

Out-of-Sample Forecast Generation

A second issue addressed in this revision is information leakage in the forecaster supplying the RL environment. Table 2's ensemble comparison is leak-free by construction (models trained only on the training partition, evaluated on the untouched test partition). However, the forecast values previously used to drive the RL environment and all downstream scheduling evaluations came from a Random Forest refit on the full dataset — including, for any given day, information from that same day's neighbourhood in the training data — which is a form of leakage distinct from the ensemble comparison in Table 2. We replaced this with a walk-forward, expanding-window forecast: an initial model is fit on the first 365 days (8 Jul 2019–6 Jul 2020, retained in-sample as an unavoidable warm-up covering 18.2% of the series) and then refit every 30 days on all data strictly before the refit point, forecasting only the following 30 days out-of-sample, for 55 refits spanning 7 Jul 2020–30 Dec 2024. For robustness testing, forecasts under perturbation during the 301-day evaluation window are recomputed with a single model trained only on data through 4 March 2024 (i.e., strictly before the test partition), avoiding any train/test contamination in the perturbed forecasts as well as the clean ones

Reinforcement Learning for Adaptive Scheduling

The best-performing ensemble model's forecast was embedded as part of the state representation used by a Deep Q-Network (DQN) agent implemented as a custom Gymnasium environment. Formally: at decision epoch i , the agent observes state s_i = the standardized feature vector at day i (46 features: demand-signal, calendar, operational and weather features, identical to the ensemble forecaster's inputs) and selects a discrete action $a_i \in \{0, 1, 2, 3, 4\}$, the number of doctors nominally proposed for day $i+1$. The realised staffing level dok_i is a as clipped by hard constraints evaluated on day $i+1$'s calendar (0 on Sundays/national holidays; ≤ 1 during collective leave; otherwise unclipped up to 4). The transition to the next state, s_{i+1} = the feature vector at day $i+1$, is fixed by the data and does not depend on a_i : the environment is a deterministic, exogenous sequence of states through which the agent moves regardless of its choices. Each training episode spans one non-overlapping pass through the training window (1,201 decision epochs, 9 Jul 2019–21 Oct 2022); consecutive episodes reset to the start of that window. Let $pred_i$ denote the ensemble-forecasted demand for day $i+1$ (produced from state s_i) and $cap_i = dok_i \times 15$ the resulting scheduled capacity. The reward function is defined as:

$$R_i = -[w_u \cdot \max(0, pred_i - cap_i) + w_o \cdot \max(0, cap_i - pred_i)] - P_0 \cdot \mathbb{I}[dok_i = 0] \dots\dots\dots (1)$$

with understaffing weight $w_u = 15$, overstaffing weight $w_o = 3$, and a fixed penalty $P_0 = 50$ applied when the agent schedules zero doctors on a regular operating day. On Sundays and national holidays, a reward of 0 is given for correctly scheduling zero doctors and -100 otherwise. The asymmetric weighting ($w_u > w_o$) reflects the higher operational cost of service shortages relative to overstaffing in a primary healthcare context; the capacity constant of 15 patients per doctor and the maximum of 4 doctors are empirically grounded, as detailed below.

Because Section “Comparison with an Exact Cost-Optimal Baseline” shows that a_i has no influence on s_{i+1} or on any future reward, the discount factor γ plays no substantive role here: the return from epoch i onward

* Corresponding author



does not depend on a_i beyond epoch i 's own reward, so the problem that DQN is asked to solve is, in this formulation, equivalent to 1,201 independent single-step decisions rather than a temporally coupled control problem. The agent (MlpPolicy, two hidden layers of 256 units, replay buffer size 50,000, batch size 64, learning rate 1×10^{-4} , discount factor $\gamma = 0.99$) was trained for 200,000 timesteps in the reward configuration reported as the primary result, and for 350,000 timesteps with a rescaled reward ($\times 0.05$) and increased exploration (exploration fraction = 0.2) in a supplementary configuration described below; four independent seeds (42, 7, 123, 2024) were run for the primary configuration.

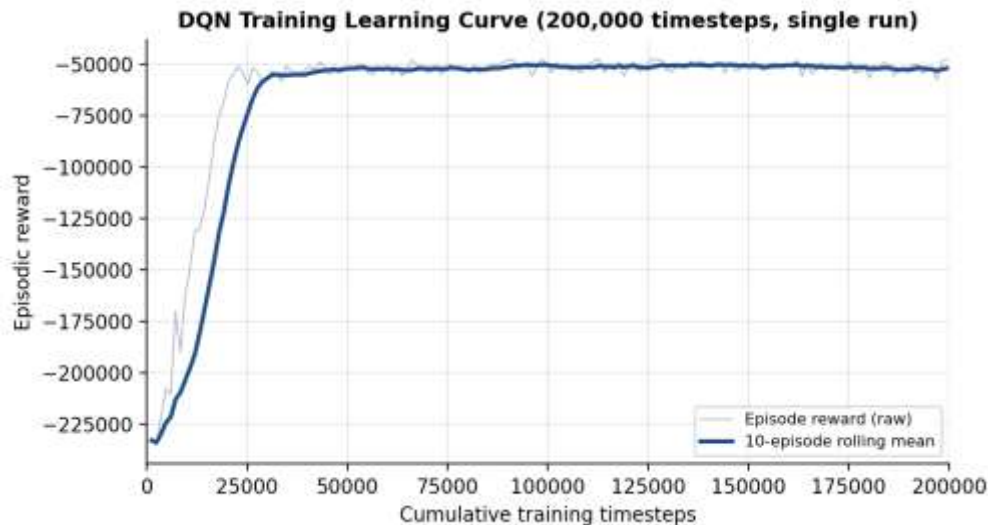


Figure 1. DQN training learning curve, seed 42, single continuous run (200,000 timesteps; 199,366 of 200,000 timesteps are covered by completed episodes, the remainder being the final in-progress episode at training cutoff)

Doctor Availability and Capacity Assumptions

The service-capacity constant of 15 patients per doctor is an observed operational quantity, not an assumption: across the 246 open target days of the held-out evaluation period, the facility recorded a mean of 51.1 patient visits per open day against a mean of 3.68 available doctors, giving an observed workload of exactly 15.0 patients per available doctor. The maximum action of 4 doctors matches the facility's historical maximum staffing level (total_dokter never exceeded 4 across the full 2019–2024 record; 2 doctors is the modal day, occurring on 58% of days, with 4 doctors available on only 8.6% of days). This bound constrains the aggregate ceiling of the action space but does not encode day-specific availability, which fluctuates independently; a policy that frequently selects the maximum action can therefore be feasible in aggregate (total doctor-days within the total ever available) while being infeasible on specific days when fewer than 4 doctors were actually on staff. We treat this as an explicit limitation (see Discussion) rather than build day-specific availability into the action space, which would require a materially different environment design.

Baseline Scheduling Policies

Four scheduling policies were compared under identical conditions, identical hard constraints, and the corrected day alignment. (i) The rule-based baseline reproduces the facility's existing practice: two doctors on regular weekdays, one on Saturdays and during collective leave, none on Sundays and national holidays. (ii) The constraint-aware forecast-to-staffing rule assigns $\lceil \text{pred}_i / 15 \rceil$ doctors (clipped by the

* Corresponding author



same hard constraints), the smallest staffing level covering the forecast; this isolates the value of the forecast itself from the value of any learned policy. (iii) The myopic cost-optimal scheduler exhaustively enumerates all five admissible actions each day and selects the one minimising exactly the reward-function cost the DQN is trained on, given the same forecast; because the objective contains no cross-day coupling (Section “Reinforcement Learning for Adaptive Scheduling”), this enumeration returns the exact per-day optimum, equivalent to solving the corresponding integer program to optimality, and constitutes the performance ceiling attainable from the forecast under the stated objective. (iv) The DQN policy. Comparing (iii) and (iv) isolates whether the learned policy adds value beyond exact optimisation of the same forecast and objective.

Evaluation

All four policies were evaluated by forward simulation over the full 301-day held-out test partition (5 March–30 December 2024), which is the chronologically final 15% of the data, disaggregated by calendar quarter and reported for the full 2024 calendar year to expose the policies to differing seasonal demand regimes; the original 60-day window used in the first submission is no longer used as a primary result, since it covers only December–early November and does not represent the annual cycle. Evaluation indicators are MAE and RMSE of scheduled capacity against actual next-day visits, shortage and overstaff day counts, total scheduled doctor-days, workload intensity (patients per scheduled doctor), and schedule volatility (mean absolute day-to-day change in scheduled doctors). Because MAE weights over- and under-staffing symmetrically while the reward function does not, we additionally report each policy's total realised cost under the reward function's own weighting (Equation 1), which is the metric each optimizing policy is actually trying to minimise. Robustness was assessed by perturbing the six demand-signal features with Gaussian noise, spikes, and their combination across ten magnitudes (0–5 standard deviations), each averaged over five random seeds, with the mean absolute perturbation actually applied logged alongside the outcome.

RESULT

Patient Visit Prediction Performance

Table 2 presents the test-set performance of the six ensemble models. Random Forest achieved the lowest error and the highest coefficient of determination ($MAE = 7.08$, $RMSE = 10.44$, $R^2 = 0.82$), outperforming the other five models.

Table 2. Test-set performance of six ensemble learning models for patient visit prediction (next-day target)

Model	MAE	RMSE	R^2
Random Forest	7.08	10.44	0.82
LightGBM	7.90	10.99	0.80
Stacking	7.81	10.68	0.81
XGBoost	7.86	11.02	0.80
Gradient Boosting	8.19	11.33	0.79
AdaBoost	8.21	11.94	0.77

Table 3 compares training and testing R^2 to assess overfitting. An earlier version of this manuscript stated

* Corresponding author



[Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.](https://creativecommons.org/licenses/by-nc-sa/4.0/)

that Random Forest had the smallest train–test R^2 gap; this was incorrect and has been removed. As the table shows, AdaBoost ($\Delta R^2 = 0.069$) and Gradient Boosting (0.128) both show a smaller gap than Random Forest (0.137). Random Forest was selected instead on the basis of superior absolute test-set performance — the lowest MAE and RMSE and the highest R^2 among all six models (Table 2) — which is the criterion that determines forecast quality for the downstream scheduling task, rather than the size of the train–test gap considered alone.

Table 3. Training vs. testing R^2 comparison across ensemble models

Model	R^2 (Train)	R^2 (Test)	ΔR^2
AdaBoost	0.837	0.768	0.069
Gradient Boosting	0.920	0.791	0.128
Random Forest	0.960	0.823	0.137
Stacking	0.958	0.815	0.143
XGBoost	0.989	0.803	0.186
LightGBM	0.991	0.804	0.187

Comparison with an Exact Cost-Optimal Baseline

Table 4 compares the four scheduling policies over the 301-day held-out period, reporting both MAE (symmetric) and total realised cost under the reward function's own asymmetric weighting (Equation 1), which is the objective each optimizing policy is actually trying to minimise. The myopic cost-optimal scheduler attains both the lowest MAE (8.11) and the lowest realised cost (9,455) of any policy, including the DQN. The DQN policy (primary seed-42 configuration, 200,000 timesteps) incurs a realised cost of 12,610 — 33% higher than the exact optimum — despite being trained to minimise exactly that quantity. This is the central finding of this revision: once information leakage and the forecast-to-decision-day misalignment are corrected, the DQN policy does not demonstrate added value over exact per-day optimisation of the same forecast, even measured on its own training objective.

Table 4. Comparison of scheduling policies over the 301-day held-out period, including the objective-consistent realised cost

Policy	MAE	RMSE	Shortage Days	Overstaff Days	Doctor-days	Realised cost (Eq. 1)
Rule-based baseline (existing)	20.61	25.71	230	13	442	88,320
Constraint-aware forecast rule	8.42	12.33	78	163	898	10,045
Myopic cost-optimal (exact IP)	8.11	11.88	93	146	871	9,455
DQN (seed 42, 200k steps)	10.06	14.13	75	170	932	12,610

Reward Rescaling, Extended Training, and Seed Repetition

Because the initial DQN result could plausibly reflect insufficient training rather than a structural limitation, two further legitimate remediation attempts were made before accepting the finding. First, the reward was rescaled by a constant factor ($\times 0.05$) to keep gradient magnitudes in a numerically better-conditioned range, a standard DQN stabilisation practice, and exploration was increased (exploration_fraction 0.1 $\rightarrow$ 0.2); this configuration, trained for 350,000 timesteps, improved MAE modestly to 9.90 but remained above the exact

* Corresponding author



[Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.](https://creativecommons.org/licenses/by-nc-sa/4.0/)

optimum. Continuing training on the seed-42 agent to 500,000 timesteps did not improve further (MAE 10.22) and increased the share of days on which the agent selected the maximum action from 76% to 96% of open days, consistent with the policy converging toward an increasingly conservative, near-constant “staff at maximum” rule rather than approaching the day-specific precision of the exact optimizer. Second, four independent random seeds (42, 7, 123, 2024) were each run for the same 350,000-timestep rescaled configuration; MAE was reasonably stable across seeds (mean 10.11, SD 0.14, range 9.90–10.26; Table 5) but the shortage/overstaff composition varied substantially between seeds (69–160 shortage days across seeds for similar MAE), indicating that different training runs converge to qualitatively different policies of similar aggregate accuracy — a further symptom of the optimization landscape lacking a single, well-shaped optimum for DQN to converge to in this formulation

Table 5. DQN performance across independent random seeds (reward-scaled configuration, 350,000 timesteps, 301-day evaluation)

Seed	MAE	RMSE	Shortage Days	Overstaff Days
42	9.90	14.28	82	162
7	10.26	14.79	151	86
123	10.22	13.90	69	176
2024	10.06	14.44	160	77
Mean $\pm$ SD	10.11 $\pm$ 0.14	—	—	—

Extended and Seasonal Evaluation

Table 6 disaggregates the comparison across calendar quarters and the full 2024 year. The ordering established in Table 4 — cost-optimal best, DQN behind both the exact optimizer and the constraint-aware rule — holds in every quarter without exception, indicating the finding is not an artefact of the particular 301-day window chosen

Table 6. Seasonal and extended evaluation (MAE) of all four policies

Evaluation period	Days	Rule-based	Constraint-aware rule	Cost-optimal	DQN
Q1 2024 (Jan–Mar)	91	21.32	10.45	9.36	11.33
Q2 2024 (Apr–Jun)	91	22.08	8.60	8.70	9.91
Q3 2024 (Jul–Sep)	92	20.60	7.87	7.48	10.17
Q4 2024 (Oct–Dec)	91	19.91	8.93	8.18	10.66
Full year 2024	365	20.98	8.96	8.43	10.52
Held-out test partition	301	20.61	8.42	8.11	10.06

Robustness Analysis

Table 7 reports the graded noise sweep using the DQN policy from the extended-training, reward-scaled configuration (seed 42, 500,000 timesteps — the same run discussed above, distinct from the 350,000-timestep seeds in Table 5). Because this policy already selects the maximum action on most open days, it is close to input-invariant by construction at low-to-moderate noise — MAE is essentially flat through 2.0 standard deviations — and only degrades materially beyond 3.0 standard deviations. This apparent robustness should be read together with Table 4: a policy that mostly ignores its input in favour of a near-constant high action is trivially stable to input perturbation, which is a different property from the policy

* Corresponding author



being well-calibrated. The spike-only condition remains largely uninformative for the reason identified in the previous round of review: at a 5% daily spike probability, mean perturbation reaching the agent never exceeds 0.2 standardised units even at nominal magnitude 5.

Table 7. Robustness under graded perturbation of the demand-signal features (301-day window, mean of five seeds)

Perturbation (SD)	Gaussian MAE	Δ vs baseline	Combined MAE	Spike MAE	Mean perturbation applied
0.00 (baseline)	10.22	—	10.22	10.22	0.00
1.00	10.18	−0.4%	10.19	10.22	0.80
2.00	10.23	+0.1%	10.22	10.22	1.60
3.00	10.55	+3.2%	10.47	10.23	2.39
5.00	10.87	+6.3%	10.81	10.33	3.99

Staffing Feasibility and Workload

The DQN policy schedules more total doctor-days over the 301-day window than were ever simultaneously available in the facility's attendance record: 932 doctor-days for the primary configuration (Table 4) and 955 for the extended-training configuration used in the robustness sweep (Table 7), against 836 doctor-days actually available across the same 246 open target days. Because the action space bounds only the daily maximum (4, matching the historical peak) and not day-specific availability — which was 2 doctors on the majority of days and 4 doctors on only 8.6% — a policy that frequently selects the maximum action, as both the DQN and, to a lesser extent, the cost-optimal scheduler do (Section “Comparison with an Exact Cost-Optimal Baseline”), can be feasible in aggregate while being infeasible on specific days. We report this as a genuine limitation of the current action-space design rather than resolve it by further tuning, since a correct fix requires day-specific availability as an input to the constraint, which is a materially different environment (see Discussion)

DISCUSSIONS

The central methodological finding of this revision is that a well-calibrated forecast, combined with exact per-day optimisation, outperforms a trained DQN policy on the same objective, and does so consistently across four random seeds, an extended and reward-rescaled training configuration, and every quarter of the evaluation year. We trace this to the environment's structure rather than to an implementation deficiency in the DQN: because the transition to the next state does not depend on the current action (Section “Reinforcement Learning for Adaptive Scheduling”), the discounted-return objective that DQN is built to exploit collapses to the same-epoch reward, and the problem is, in substance, 1,201 independent per-day decisions rather than a sequential control task. Under that structure, an algorithm that must approximate a value function through temporal-difference bootstrapping has no advantage over — and in our results, converges to a worse solution than — direct enumeration of the five-action space against the known forecast each day. This is consistent with the broader observation that RL's value over supervised or exact-optimisation baselines is concentrated in problems with genuine delayed consequences (queue build-up, resource contention across periods, fatigue accumulation) (Sutton & Barto, 2018), which the present environment, as specified, does not contain.

This reframes the comparison against prior RL-for-scheduling studies (Lee & Lee, 2020; Zuo et al., 2024; Liu et al., 2025) less as “this study extends RL to a new setting” and more as a cautionary methodological point: those studies' environments plausibly retain cross-period coupling (e.g., a patient queue that carries

* Corresponding author



[Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.](https://creativecommons.org/licenses/by-nc-sa/4.0/)

over, or resources contended across simultaneous cases) that a single-facility, single-resource, next-day capacity match does not. Authors proposing RL for operational healthcare scheduling should establish, as this study did only after the fact, whether the environment as formulated actually contains action-dependent state transitions before expecting RL to outperform an exact or greedy baseline; where it does not, the myopic optimum is cheaper to compute, easier to audit, and — in this study — more accurate.

The corrected results also qualify the earlier robustness and workload analyses. The DQN policy's apparent noise-robustness (Table 7) is largely attributable to its converging toward a near-constant maximum-staffing policy rather than to genuine stability of a nuanced decision rule; a policy that mostly ignores its input cannot be meaningfully destabilised by perturbing that input. Similarly, the earlier claim that DQN uses staffing resources more efficiently than an uplifted rule-based schedule no longer holds under the corrected setup: DQN now uses more doctor-days than the exact optimizer while achieving worse accuracy, and its aggregate staffing level exceeds the facility's historically available doctor-days, a feasibility issue that a purely aggregate action-space bound (Section “Doctor Availability and Capacity Assumptions”) does not prevent.

Several limitations remain. First, the study draws on a single primary healthcare center; generalisation to facilities with different demographics, staffing structures, or genuine cross-period coupling (e.g., multi-day shift patterns, contracted minimum hours) is untested and, per the discussion above, may change which method is preferable. Second, the action space bounds only the aggregate daily maximum and not day-specific doctor availability, so aggregate feasibility does not guarantee day-level feasibility; incorporating a day-specific availability constraint is identified as necessary future work rather than resolved here. Third, robustness evidence is now understood to partly reflect policy degeneracy rather than calibrated stability, and should be re-examined for any future environment formulation that restores genuine sequential structure (for instance, a weekly staffing budget or fatigue constraint linking consecutive days), which would also be the natural way to give reinforcement learning a legitimate structural advantage over exact per-day optimisation in this domain. Finally, all policies were evaluated in forward simulation against recorded demand; a prospective operational trial has not been conducted.

CONCLUSION

This study built a five-year, leak-free, day-aligned forecasting-to-scheduling pipeline for a primary healthcare center in Palembang, Indonesia, and used it to test — rather than assume — whether a Deep Q-Network scheduling agent adds value beyond direct optimisation of an accurate demand forecast. Random Forest provided the best next-day visit forecast among six ensemble models (MAE = 7.08, RMSE = 10.44, $R^2 = 0.82$). Once two methodological issues were identified and corrected — information leakage in the forecaster driving the scheduling environment, and a one-day misalignment between the forecast target and the day against which it was evaluated — a myopic, exact per-day optimizer solving the DQN's own objective outperformed the trained DQN policy on every evaluation window tested, across four random seeds and two training configurations, and even on the DQN's own training objective (realised cost 9,455 for the exact optimizer versus 12,610 for DQN). We attribute this to the environment's lack of action-dependent state transitions, which removes the theoretical basis for reinforcement learning's advantage over exact optimisation. The contribution of this study is accordingly a rigorously validated, reproducible forecasting-to-scheduling pipeline, together with an empirically grounded diagnosis of when the added complexity of reinforcement learning is, and is not, justified for demand-responsive capacity allocation in primary healthcare — a question we believe is more useful to the field, answered honestly, than an unexamined claim of RL superiority would have been. Future work should test environment formulations with genuine cross-day coupling (weekly staffing budgets, fatigue or continuity constraints), incorporate

* Corresponding author



[Creative Commons Attribution-NonCommercial-ShareAlike 4.0
International License.](https://creativecommons.org/licenses/by-nc-sa/4.0/)

day-specific doctor availability into the action space, and progress from forward simulation to a prospective operational trial.

ACKNOWLEDGMENT

The authors gratefully acknowledge the research permit issued by the Badan Kesatuan Bangsa dan Politik (Kesbangpol) of the Palembang City Government (Permit No. 070/00993/BAN-KBP/2025, dated 12 June 2025), and thank Universitas Indo Global Mandiri, the Palembang City Health Office (Dinas Kesehatan Kota Palembang), and the management of Puskesmas Talang Ratu, Palembang, for granting research access and providing the operational data used in this study.

REFERENCES

- Fatikasari, D., Dwi Pratama, Y. S., & Hozairi, H. (2024). Optimasi Penjadwalan Tenaga Kesehatan di Puskesmas Teja Kabupaten Pamekasan Menggunakan Solver Excel. *TeknoIS: Jurnal Ilmiah Teknologi Informasi Dan Sains*, 14(2), 214–224. <https://doi.org/10.36350/jbs.v14i2.257>
- Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5). <https://doi.org/10.1214/aos/1013203451>
- Fu, Y., Wang, Y., Gao, K., & Huang, M. (2024). Review on ensemble meta-heuristics and reinforcement learning for manufacturing scheduling problems. *Computers and Electrical Engineering*, 120, 109780. <https://doi.org/10.1016/j.compeleceng.2024.109780>
- Hendiawati, Y., Abdussalaam, F., & Gunawan, E. (2023). Tata Kelola Sistem Informasi Jadwal Praktik Dokter Berbasis Web dengan Menggunakan Framework Laravel. *Jurnal Teknologi Sistem Informasi Dan Aplikasi*, 6(4), 590–600. <https://doi.org/10.32493/jtsi.v6i4.33690>
- Karami, F. H. A. (2018). *Optimasi Penjadwalan Staf dengan Menggunakan Algoritma Reinforcement Learning Hyper-Heuristics Studi Kasus Rumah Sakit Ibu dan Anak Kendangsari*. Institut Teknologi Sepuluh Nopember.
- Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T. Y. (2017). LightGBM: A highly efficient gradient boosting decision tree. *Advances in Neural Information Processing Systems, 2017-Decem(Nips)*, 3147–3155.
- Lee, S., & Lee, Y. H. (2020). Improving Emergency Department Efficiency by Patient Scheduling Using Deep Reinforcement Learning. *Healthcare*, 8(2), 77. <https://doi.org/10.3390/healthcare8020077>
- Li, Y., Wang, H., Wang, N., & Zhang, T. (2022). Optimal scheduling in cloud healthcare system using Q-learning algorithm. *Complex and Intelligent Systems*, 8(6), 4603–4618. <https://doi.org/10.1007/s40747-022-00776-9>
- Liu, K., Li, X., Zou, C. C., Huang, H., & Fu, Y. (2020). Ambulance Dispatch via Deep Reinforcement Learning. *Proceedings of the 28th International Conference on Advances in Geographic Information Systems*, 123–126. <https://doi.org/10.1145/3397536.3422204>
- Liu, R., Piplani, R., & Toro, C. (2022). Deep reinforcement learning for dynamic scheduling of a flexible job shop. *International Journal of Production Research*, 60(13), 4049–4069. <https://doi.org/10.1080/00207543.2022.2058432>
- Liu, X., Zheng, C., Chen, Z., Liao, Y., Chen, R., & Yang, S. (2025). *Reinforcement Learning for Patient Scheduling with Combinatorial Optimisation* (pp. 238–243). https://doi.org/10.1007/978-3-031-77918-3_18
- Mahajan, P., Uddin, S., Hajati, F., & Moni, M. A. (2023). Ensemble Learning for Disease Prediction: A Review. *Healthcare*, 11(12), 1808. <https://doi.org/10.3390/healthcare11121808>
- Mahariani, Y. R. (2023). Penjadwalan Ruang Operasi Rumah Sakit dengan Metode Non-Dominated Sorting Genetic Algorithm II (NSGA-II). *JUPI (Jurnal Ilmiah Penelitian Dan Pembelajaran Informatika)*, 8(1), 338–344. <https://doi.org/10.29100/jupi.v8i1.3989>
- Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller,

* Corresponding author



[Creative Commons Attribution-NonCommercial-ShareAlike 4.0
International License.](https://creativecommons.org/licenses/by-nc-sa/4.0/)

- M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., & Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540), 529–533. <https://doi.org/10.1038/nature14236>
- Pratiwi, E. W., & Siambaton, M. Z. (2022). Aplikasi Penjadwalan Dokter Pada Rumah Sakit Umum Kota Pinang dengan Menggunakan Algoritma Greedy. *Hello World Jurnal Ilmu Komputer*, 1(1), 1–9. <https://doi.org/10.56211/helloworld.v1i1.4>
- Priatna, W., Warta, J., & Sulistiyo, D. (2023). Implementasi Algoritma Genetika untuk Aplikasi Penjadwalan Sistem Kerja Shift. *Techno.Com*, 22(1), 235–246. <https://doi.org/10.33633/tc.v22i1.7049>
- Stevens, C. A., Lyons, A. R., Dharmayat, K. I., Mahani, A., Ray, K. K., Vallejo-Vaz, A. J., & Sharabiani, M. T. (2023). Ensemble machine learning methods in screening electronic health records: A scoping review. *DIGITAL HEALTH*, 9. <https://doi.org/10.1177/20552076231173225>
- Wang, L., & Demeulemeester, E. (2023). Simulation optimization in healthcare resource planning: A literature review. *IIE Transactions*, 55(10), 985–1007. <https://doi.org/10.1080/24725854.2022.2147606>
- Yu, C., Liu, J., Nemati, S., & Yin, G. (2023). Reinforcement Learning in Healthcare: A Survey. *ACM Computing Surveys*, 55(1), 1–36. <https://doi.org/10.1145/3477600>
- Zuo, J., Jin, Y., & Liu, W. (2024). Outpatient scheduling problem in smart hospital with two-agent deep reinforcement learning algorithm. *Discover Computing*, 27(1), 41. <https://doi.org/10.1007/s10791-024-09474-1>