Reinforcement Learning for ICU Treatment Planning

Goureesh Chandra	31	TVE22CS069
Ivin Mathew Kurian	33	TVE22CS075
Muhammed Farhan	39	TVE22CS094
Rethin Francis	72	LTVE22CS149

Advisor: Prof. Divya S K

College of Engineering, Trivandrum
Dept. of Computer Science & Engineering

October 15, 2025

Outline of Presentation

Introduction

Doctors face tough decisions treating critically ill ICU patients with conditions like sepsis. Our AI system learns from ICU data to suggest better plans.
Using reinforcement learning, it builds a "digital twin" simulator to predict patient responses to treatments safely. This enables personalized, safer plans, supporting doctors and reducing risks/costs.

Problem Statement and Objectives

Problem: Timely, personalized ICU treatment decisions are challenging due to dynamic conditions and fragmented data.

- High-stakes, dynamic trajectories.
- Incomplete EHR data.
- AI limited to historical mimicry.

Objectives:

- Build patient simulator from EHR data.
- Optimize RL policies safely.
- Provide clinical decision support.

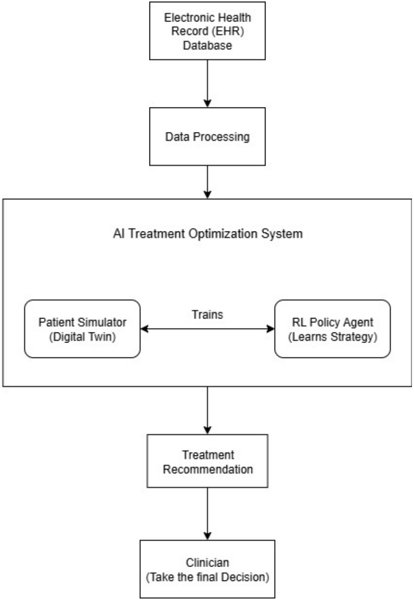
Design Methodology and Algorithms

This approach is inspired by DreamerV3 and its clinical adaptation in medDreamer.

Two sequential stages:

1. Train the **Patient Simulator (World Model)** to predict patient outcomes.
2. Train the **RL Agent (Policy)** using the simulator to discover optimal treatments.

Design Methodology and Algorithms: Diagrams



Algorithm 1: Patient Simulator Training

Goal: Learn patient health dynamics from EHR data to create a reliable "digital twin".

Steps:

1. **Initialization:** Neural network parameters ϕ
   - Encoder
   - Dynamics predictor
   - Prediction heads (reconstruction, rewards, outcomes)
2. **Training Loop:** For multiple epochs:
   1. Sample patient trajectories (observations, actions, rewards)
   2. Encode observations to latent states s_t
   3. Predict next latent state s_{t+1} given s_t and action a_t
   4. Reconstruct observations and predict rewards/outcomes
   5. Compute combined loss and update ϕ via backpropagation
3. **Output:** Trained Patient Simulator for realistic patient trajectories

Algorithm 2: RL Agent Training

Goal: Train an optimal treatment policy safely in the simulated environment.

Steps:

1. Load trained Patient Simulator (weights frozen)
2. Initialize RL Agent: policy network θ (actor), value network ψ (critic)

Phase 1: Clinically Grounded Policy Initialization

- Generate hybrid trajectories (real data + short simulator rollouts)
- Train actor and critic:
  - Critic: evaluate long-term state values
  - Actor: choose actions leading to high-value states

Algorithm 2 (cont.): Strategic Refinement

Phase 2: Strategic Refinement through Long-Horizon Imagination

- Generate fully imagined trajectories using the Patient Simulator
- Fine-tune actor and critic to improve long-term planning

Output: Final trained **Optimal Treatment Policy** that recommends actions given a patient's current state

Implementation Setup Framework

4.1 Hardware and Software Requirements

- **Hardware:** GPU-enabled system (RTX 4060/4090 or cloud GPUs).
- **Software:** Python, PyTorch, Ray RLlib, pandas, NumPy, scikit-learn.

4.2 Implementation Environment Setup

- Datasets: MIMIC-III, MIMIC-IV, eICU.
- Preprocessing: Clean and structure EHR data into trajectories.
- Frameworks: PyTorch for neural networks, Ray RLlib for RL.

Implementation Results

Phase-1 Results: As this is Phase-1, full implementation is ongoing. Preliminary results include:

- Data preprocessing completed for MIMIC-IV demo.
- Patient Simulator training initiated (e.g., loss curves placeholder).
- Synthetic FHIR Bundle generation for data validation.

Graphs/Tables:

- Example: Training loss over epochs (insert graph if available).
- Screenshots: Model architecture diagram.

Individual Contributions

- **Goureesh Chandra (TVE22CS069):** Literature review, algorithm design, and data preprocessing.
- **Ivin Mathew Kurian (TVE22CS075):** Model implementation, simulator training, and notebook development.
- **Muhammed Farhan (TVE22CS094):** RL agent training, evaluation, and presentation preparation.
- **Rethin Francis (LTVE22CS149):** Data handling, FHIR mapping, and documentation.

Future Works (Project Phase-2)

- Complete full RL training and evaluation on larger datasets.
- Extend framework to other diseases (e.g., diabetes, cancer care).
- Incorporate multi-modal data (genomics, imaging).
- Clinician-in-the-loop evaluation and real-world validation.

Conclusion

- We are developing a **model-based reinforcement learning framework** trained on ICU data for clinical decision support.
- By building a **digital twin of patients**, the system safely explores treatment strategies without direct risk to patients.
- The approach enables **personalized and safer treatment recommendations**, reducing unnecessary interventions and improving patient outcomes.
- Future work includes extending the model to more diverse patient populations and validating in real-world clinical settings.

References

Qianyi Xu, Dilruk Perera, Gousia Habib, and Mengling Feng (2025).
medDreamer: Model-Based Reinforcement Learning with Latent Imagination on Complex EHRs for Clinical Decision Support.
arXiv preprint arXiv:2505.19785.

Arthur Komorowski et al. (2021).
Challenges with reinforcement learning model transportability for sepsis treatment in emergency care.
Critical Care Medicine.

Ali Amirahmadi, Mattias Ohlsson, and Kobra Etminani (2023).
Deep learning prediction models based on EHR trajectories: A systematic review.
Journal of Biomedical Informatics.

Mila Nambiar et al. (2023).
Deep offline reinforcement learning for real-world treatment optimization applications.
arXiv preprint arXiv:2302.07549.

Anon. Authors (2023).
Mastering Memory Tasks with World Models.
ICML.

Edward De Brouwer, Javier González Hernández, and Stephanie Hyland (2022).
Predicting the impact of treatments over time with uncertainty aware neural differential equations.
AISTATS.

Flemming Kondrup et al. (2023).
Towards Safe Mechanical Ventilation Treatment Using Deep Offline Reinforcement Learning.
AAAI-23.

Thank You!