

Pressure Prediction in Mechanical Ventilators Using Neural Networks: Workshop 3

Cristian Parroquiano, Juan González, Santiago Chavarro, Joel Pérez

Department of System Engineering
Universidad Distrital Francisco José de Caldas
Bogotá, Colombia

Codes: 20222020192, 20222020200, 20231020219, 20242020017

Abstract—The purpose of this workshop is to develop a robust design for the ventilator pressure prediction system using clever strategies to link every component of the system, such as mathematical equations and tools to manage the asynchronous data, with this being able to understand how the flow of data can be implemented [1].

Index Terms—Mechanical ventilators, neural networks, pressure prediction, LSTM, system implementation, simulation, medical devices, chaos mitigation

I. INTRODUCTION

The problem encompasses a significant inconvenience: every ventilator's pressure needs to be changed manually for every patient, being a process that demands excessive resources. To solve this problem, various researchers and groups have designed alternatives, such as lung simulators, sensors, and mathematical implementations [2]. The purpose of our investigation is to contribute our potential solution to this challenge. We begin by summarizing all the steps the system takes from input to output and feedback.

The system starts with the input of air, which passes through different valves that regulate pressure, filtration, and various emergency scenarios that might occur within the system. The risks we may encounter vary between significant data randomness, improper manipulation of the physical tool, and malfunctions in the pressure delivered to the lung [3].

With that, the core objective is to implement a neural network to achieve the system's needs. Through robust design, the system can be better understood, making implementation clearer and easier to follow, with the help of socket strategies, differential equations, and machine learning tools [4].

II. METHODS AND MATERIALS

The first method we aim to design is the behavior of the neural network. We designed the tool to use TensorFlow, a Google open-source library that emphasizes machine learning and deep learning, allowing asynchronous data to be processed [1].

A differential equation was designed to calculate pressure over time and predict its behavior [2]:

$$P(t) = R \cdot Q(t) + \frac{1}{C} V(t) + L \frac{dQ}{dt} \quad (1)$$

where $P(t)$ is the pressure over time, $Q(t)$ is the airflow over time, $V(t)$ is the air volume over time, R is the lung flow

resistance, C is the volume compliance of the air, and L is the inductance of the pressure.

To complement the previous tools, we designed a socket behavior to intercommunicate asynchronous data. With sockets, the data for the required patient pressure and the incoming air data communicate during the inhalation and exhalation process. This was designed to solve the problem of data randomness [3].

The system architecture organizes into seven coordinated layers achieving design objectives through integrated functionality:

Data ingestion layer: Captures real-time information from multiple sources including pressure sensors, flow meters, oxygen concentration detectors, manual medical staff inputs, and patient demographic data. Incorporates range validation and consistency checks for early sensor failure detection.

Preprocessing module: Transforms raw data into model-ready features implementing digital filtering for noise reduction, adaptive normalization for sensor scale harmonization, sliding windows for temporal dependency capture, and robust statistical methods for outlier management [4].

Sockets: Implementing socket strategy to let the data communicate between the agents, the patient and the ventilator, the data goes through the sensor to modulate the pressure in real time.

Machine learning: Hybrid architecture combining LSTM neural networks with physical respiratory equations [1]. LSTMs capture complex temporal patterns and long-term dependencies, while physical R-C models ensure physiological consistency, mitigating implausible prediction risks.

Safety controller: Independent verification system ensuring clinically safe prediction ranges before actuator command transmission. Implements finite state machines for operational mode transitions and fallback logic for high-uncertainty scenarios [3].

Actuation layer: Translates controller decisions into precise inspiratory/expiratory valve signals and PEEP valve management for positive end-expiratory pressure maintenance. Incorporates position feedback for command execution verification.

Monitoring system: Captures real-time performance metrics including prediction accuracy, processing latency, and

hardware status. Generates scalable alerts from operational warnings to emergency interventions [4].

Now the following graph summarizes this architecture:

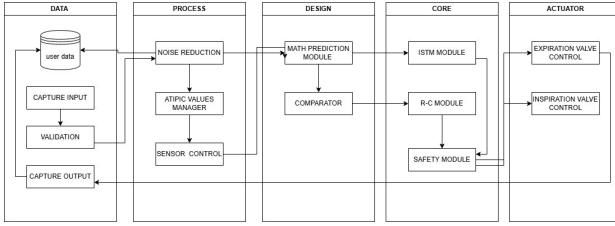


Fig. 1. System architecture showing layer interactions and feature relationships

Finally, we can describe the user interaction with the system:

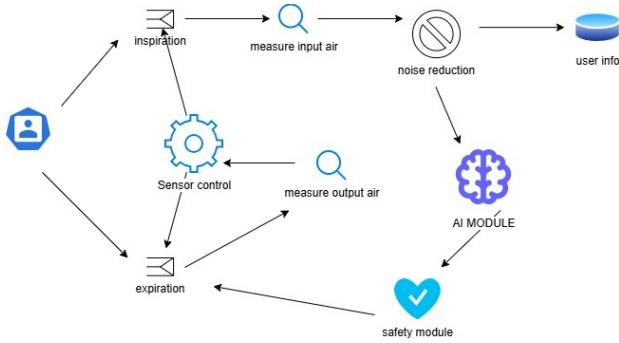


Fig. 2. System architecture showing behavior of elements

Quality and standards: Our design choices are informed by international standards to ensure quality and safety. IEC 62304 for medical device software guides our software lifecycle processes, including risk management and verification. Principles from ISO 9001 are applied to our quality management system, ensuring consistent design and development. The structured, measurable approach of CMMI and the data-driven quality focus of Six Sigma inform our process improvement and variability reduction efforts [5].

Project management plan: To develop and implement this system, we chose the Scrum method as the project management plan. Thanks to it, the development process becomes more flexible and allows for constant review of progress or increments. First, it is necessary to define the Scrum roles:

Product owner

Scrum Master

Development team

Now the next step is to define the user stories:

- As a medical operator, I want the system to capture patient respiratory data so that I can start the monitoring process
- As a system, I want to validate the input data to ensure consistency and prevent errors [2].
- As a medical operator, I want the captured data to be securely stored so that it can be used for future learning and analysis

- As a system, I want to detect and manage atypical values so that invalid or extreme readings don't affect the control logic [3].
- As an AI module, I want to use an LSTM neural network to learn from patient data so that I can adjust future predictions dynamically [1].
- As a system, I want to compare real-time measurements with predicted values to detect deviations or anomalies
- As a safety manager, I want the system to trigger safety protocols when abnormal conditions are detected so that the patient remains protected
- As a control system, I want to adjust the inspiration valve automatically based on predicted needs [4].
- As a control system, I want to adjust the expiration valve automatically to synchronize with the patient's breathing cycle.
- As a medical operator, I want to monitor both valves in real time so that I can verify their behavior and override if needed

The final step is to define a timeline for the sprints and set the deliverables:

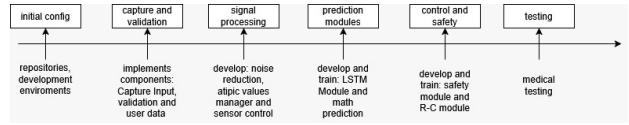


Fig. 3. System flux behavior

Risks and mitigation:

Model failure and physiologically implausible predictions: The data can be processed with close to zero relation, leaving the problem with very high randomness. To mitigate this, our hybrid LSTM-physical model ensures predictions are grounded in known respiratory physics [1]. The independent Safety Controller acts as a final verification layer, blocking any command that falls outside predefined safe pressure ranges. This is a defense-in-depth strategy.

Given that data behavior, the data is open to data loss, where some vital information is gone and may affect the function of the system. To mitigate this problem, we implement a containerized deployment with Kubernetes, allowing for automated failover and self-healing if a component fails [4]. Data persistence is managed through robust databases (TimescaleDB, PostgreSQL), and sensor data streams are handled by Apache Kafka, which is designed for high durability and availability.

Another emergency that might appear could be a security breach, where unauthorized people manipulate the ventilator. To solve this, we provide access to the system and its data to be controlled through authentication and authorization protocols [5]. Communication between components, especially between sensors and the central system, is encrypted. The system is designed to be deployed on a secure, isolated network, following best practices for networked medical devices.

Monitoring and response:

The Monitoring System is our primary tool for operational vigilance. It provides real-time KPIs on system health, latency, and prediction accuracy [2]. Alerts are configured to notify engineers immediately of any anomalies. During development, we employ a continuous integration/continuous deployment (CI/CD) pipeline with automated testing to catch issues early, and all changes are tracked and reviewed following a structured process [3].

III. RESULTS AND DISCUSSION

While discussing the tools designed, we faced a problem with the design pattern of observer. With the help of the professor, we understood that a better way to manage the data is a socket strategy, where two agents, the ventilator and the patient, communicate between the sensor and let the data be processed in a more efficient way [4].

Our proposed hybrid architecture, which fuses a data-driven LSTM network with a physics-based R-C model, is a direct response to the limitations identified in previous research [1]. Pure data-driven models risk generating physiologically implausible predictions, especially during the chaotic transitions between respiratory phases. By grounding the neural network's predictions with the foundational respiratory equation (Eq. 1), we aim to create a system that is both adaptive and inherently safe. The LSTM's strength in capturing long-term temporal dependencies is theorized to be ideal for learning a patient's unique respiratory patterns, while the physical model acts as a built-in regularizer, constraining outputs to a clinically meaningful range [2].

The multi-layered safety protocol, culminating in an independent Safety Controller, is arguably the most crucial component of our design [3]. It operationalizes the principles of standards like IEC 62304, moving beyond mere algorithmic accuracy to functional safety. This controller, with its finite state machines and fallback logic, is designed to be the final arbiter, preventing any potentially harmful command from reaching the actuators. This design choice highlights our prioritization of patient safety over pure predictive performance.

However, we acknowledge significant challenges that lie ahead. The most prominent is the reliance on simulated data for initial development [1]. The dataset's scarcity and lack of pathological diversity mean that our model's ability to generalize to real-world patients with conditions like COPD or ARDS remains unproven. Furthermore, the mitigation strategies for sensitivity and chaos, while sound in theory, require extensive validation under clinical conditions to fine-tune their parameters and confirm their efficacy [4].

Looking forward, the success of this system hinges on a rigorous validation pipeline. This must include not only testing on more diverse and realistic datasets but also Hardware-in-the-Loop (HIL) simulations to verify the integration of software and hardware under real-time constraints. The ultimate test will be a side-by-side comparison with existing PID-controlled ventilators across a range of performance metrics, including accuracy, response time, and stability,

to quantitatively demonstrate the advantages of our neural network-based approach [2].

IV. CONCLUSION

We delivered a modular, resilient architecture that fuses temporal learning with physics constraints and a safety controller to ensure clinically plausible pressure predictions under real-time constraints [1]. The design incorporates redundancy, health checks, circuit breakers, and latency targets, with monitoring that surfaces accuracy and reliability KPIs for continuous improvement [4]. Looking forward, systematic validation on diverse synthetic patient profiles, sensitivity analysis of safety thresholds, and hardening of deployment practices will consolidate the system's readiness for the final project phase [3].

REFERENCES

- [1] S. Diao, J. Wang, and Others, "Ventilator pressure prediction using recurrent neural network," *arXiv preprint*, 2024. [Online]. Available: <https://arxiv.org/abs/2410.06552>
- [2] V. Authors, "Development of artificial neural networks for the prediction of the pressure," *Flow Measurement and Instrumentation*, 2024. [Online]. Available: https://www.academia.edu/126593545/Development_of_artificial_neural_networks_for_the_prediction_of_the_pressure
- [3] A. of the Article, "A new method for pore pressure prediction on malfunctioning cells using artificial neural networks," *Water Resources Management*, vol. 35, pp. 979–992, 2021. [Online]. Available: <https://link.springer.com/article/10.1007/s11269-021-02763-0>
- [4] V. Researchers, "A review of finite element analysis and artificial neural networks," *Materials*, vol. 14, no. 20, p. 6135, 2021. [Online]. Available: <https://pmc.ncbi.nlm.nih.gov/articles/PMC8538846/>
- [5] I. P. Roberts, "Ieee bibliographies in latex," 2020. [Online]. Available: https://ianroberts.gitlab.io/notes/ieee_bibliography.html