Acute Kidney Injury (AKI) Detection System

System Design Document

Authors:
Cesare Bergossi, Ginevra Cepparulo, Asia Belfiore, Lisa Faloughi

Computing Department
Imperial College London

January 31st, 2025

Objective

We aim to build a real-time *Acute Kidney Injury* (AKI) detection system for *South Riverside Hospital* that will automatically predict patients' AKI risk by instantaneously processing patients' blood test results. If AKI is detected, the system will immediately alert the Hospital's Clinical Response Team through their existing Pager System.

Motivation

AKI is a quickly developing condition and can be the result of many serious other medical ailments. It is generally detected by monitoring the levels of creatinine in the blood, which can signal presence of kidney damage. South Riverside Hospital does not currently provide an automated AKI detection system, as doctors have to manually review test results, entailing a higher risk of incorrect prediction due to human error and lengthy delays. Our system represents an automated solution that could lead to faster and more reliable AKI detection, thus improving hospital workflows and reducing the strain on the Hospital's clinical staff.

Goals

- Real-time **processing** of incoming MLLP/HL7 messages from the Hospital's systems within minimum latency.
- Handling of up to 20 daily patient **admissions** or **discharges** and up to 100 blood **test analyses** per day.
- Average **throughput** of 0.0027 tests/sec, in order to withstand the analysis of up to 100 tests per day, assuming bursts of incoming test results during peak hours within 8am-8pm
- Automatic **analysis** of new blood test results combined with patients' historical data to detect potential AKI cases.
- Immediate **alerting** of the Hospital's Clinical Response Team for detected AKI cases within 3 seconds from receival of HL7 message with blood test result.
- Ensured seamless **integration** with the hospital's infrastructures.

Non Goals

- **Doctor Replacement**: it only serves as a detection and alerting system, it will not recommend treatments or medications.
- **Redesign the hospital's IT infrastructure**: it will work with existing hospital systems.
- **Beyond AKI detection**: it will not deal with treatments or other illnesses.

Outline Solution

To automate AKI detection and alert doctors in real-time, our system will listen for patient test results, store them and use them, alongside historical clinical data, to predict AKI leveraging our pre-trained Machine Learning model, triggering immediate alerts whenever AKI is detected.

Background

Our system will be listening to a single real-time stream of messages generated independently by the Hospital's *Patient Administration System* (PAS) and *Laboratory Information Management System* (LIMS). The PAS sends two types of messages: patient admission and discharge logs from the hospital. The LIMS sends creatinine levels from patients' blood test results.

Both the PAS and the LIMS systems generate standard HL7v2 messages. *Minimal lower layer protocol* (MLLP) is employed in order to allow the transmission of multiple messages across a single stream. The transmission of HL7 messages happens over a TCP/IP connection. The staff paging alerting happens as a HTTP post.

Proposed Design

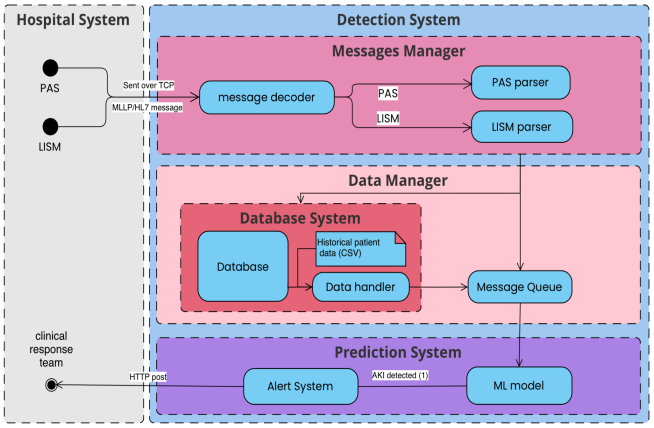

Figure 1a: System Components and Interaction with Hospital Systems

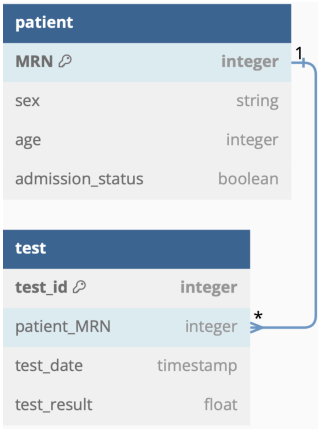

Figure 1b: Database

Figure 1: Detection System Specifications (Figure 1a) and Database Structure (Figure 1b).

1. **Messages Manager**: handles messages received from the hospital system, ensures they're not corrupted and handles message decoding and data extraction. It builds upon two components:
   - **Message Decoder**: Listens for incoming messages on a TCP socket, receives MLLP messages and parses them into separate PAS and LISM messages in HL7v2 format.
   - **PAS Parser:** Receives PAS messages as HL7, extracts patient data (MID, admission/discharge date) and makes a call to update the database's patient table.
   - **LISM Parser**: Receives LISM message as HL7, extracts blood test results data (MID, creatinine levels, test date) and makes a call to update the databas's test table.

2. **Data Manager:** receives pre-processed data from the Messages Manager, stores it and queries for AKI predictions as soon as new blood test results are received
   - **Database System**: handles patient data. It is made of the following submodules:
     - **Database**: stores patient data (MID, sex, age, hospital admission/discharge status) and patient blood test data (MID, history of test dates and results) in separate tables (Figure 1b). The two tables are related by the common 'MID' key. The patient data is updated differently based on the patient's admission status and presence of historical data (i.e. patient has been previously addìmitted to the Hospital or not). The blood test results table is updated by first checking whether the patient table contains the required information about the patient associated with that blood test result.
     - **Data Handler**: Queries patient data from the database and (if present) retrieves their historical data from a *Source CSV File* and uses it to trigger the Message Queue component once the system has successfully received both PAS and LISM data for a given patient.
   - **Message Queue**: keeps track of the incoming patient updates and, when triggered by the Data Handler, queries for AKI predictions from the Prediction System.

3. **Prediction System:** predicts the AKI diagnosis for given patients and alerts the clinical response team if AKI is detected
   - **ML Model:** The AKI Predictor processes the patient data received, extracts relevant patient features and predicts an AKI risk score using a trained XGBoost model. The ML model achieves an F3 score of (0.997), substantially higher than the current standard NHS detection algorithm (which has an F3 score of 0.73). This means that our model has a lower tendency to misclassify sick individuals as healthy when compared to the standard algorithm.
   - **Alert System**: responsible for sending an alert to the Hospital's Pager System through a HTTP post, notifying the appropriate clinical response team if AKI has been detected in a given patient. It also ensures the successful reception of the alert.

Alternatives Considered

Approach One: Batch Processing of Test Results (Rejected)

In this approach, the system would collect patient test results over a fixed time-window (e.g. hourly or daily) and analyze them in bulk, generating a report for hospital staff to review. This was rejected due to the problematic latency in AKI detection, where high-risk patients could go unnoticed for hours, thus posing a risk for delayed treatment.

Approach Two: Manual Review with AI Assistance (Rejected)

This approach proposed a hybrid system where an AI model would be used to flag high-risk cases but a human reviewer, such as a nurse or doctor, would be required to verify predictions and to manually trigger an alert in detected cases. It was rejected due to the slow response times (as alerts could be delayed due to staff availability constraints) and increased workload introduced by the manual verification step.

Testing

- **Unit Testing**: Each system component will be tested in isolation, including their edge cases, to ensure proper functionality. The Message Manager will be checked for accurate HL7 parsing and data extraction; the ML Model for correct classification, and the Alert System for proper message construction and delivery.
- **Integration Testing**: End-to-end tests using mock HL7 messages and mock/test data will simulate real hospital workflows to ensure the system behaves as expected.
- **Performance Testing**: We will employ latency, scalability and recovery tests to ensure the system is able to withstand heavy data streams accurately and in a timely manner, and to adequately recover from unexpected System Crashes.

Future Work

- **Risk Severity Alert**: Instead of a one-size-fits-all approach, alerts could be prioritized by severity, and routed directly to the right specialist.
- **Additional Features**: Beyond detection, the system could provide decision-support features, suggesting the appropriate next steps based on medical guidelines and past case studies.
- **Multi-Structure Compatibility**: The system could be standardized to support different hospital infrastructures, deployed via cloud-based solutions.