A Machine Learning System for AKI Prediction

Objective

The objective is to develop a system that allows the current South Riverside hospital communication standards to interface seamlessly with our developed acute kidney injury detection model. The system will process a real time stream of messages from the hospital consisting of the times when patients entered or left the hospital and the results of their blood tests. Using this information and our detection model, we will predict whether a patient has developed an acute kidney injury or not. In the case of a positive result, we shall alert the hospital's clinical response team by making a request to the hospital's pager management system with the patient's details. Our system will be designed in such a way to maximize throughput and minimize latency.

Motivation

According to the NHS, 'It is estimated that one in five emergency admissions into hospital are associated with acute kidney injury (1), that up to 100,000 deaths in secondary care are associated with acute kidney injury and that 1/4 to 1/3 have the potential to be prevented'. Furthermore, it is estimated that the additional cost upon the healthcare economy is £500 million (2). Looking at these statistics, it is clear that a substantial improvement to the testing procedure has immense potential to lessen the human toll and economic burden of acute kidney injury. With our new system, we intend to work towards this goal by giving doctors a more reliable prediction method and a headstart in treating the condition.

Goals

- Build reliable software for pre-processing the PAS and LIMS system messages
- Use our acute kidney injury detection model to predict whether the particular patient has developed the condition with an f3 score greater than 0.73
- Build reliable software for making a request to the hospital's pager management system in the case of a positive result
- Our system should be able to process significantly more than 20 admissions (and therefore discharges) and around 100 creatine blood results per day
- The time between receiving a HL7 message with a creatinine result, and paging the clinical response team should the result indicate acute kidney injury, should be under 3 seconds.

Non-goals

This design document focuses specifically on the real-time data pathway for model interference and alerting. As such, the following areas are considered non-goals:

- Modifying or retraining the underlying AKI model
- Determining the severity of AKI and recommendation of the next course of action
- Building user interfaces for model explanations

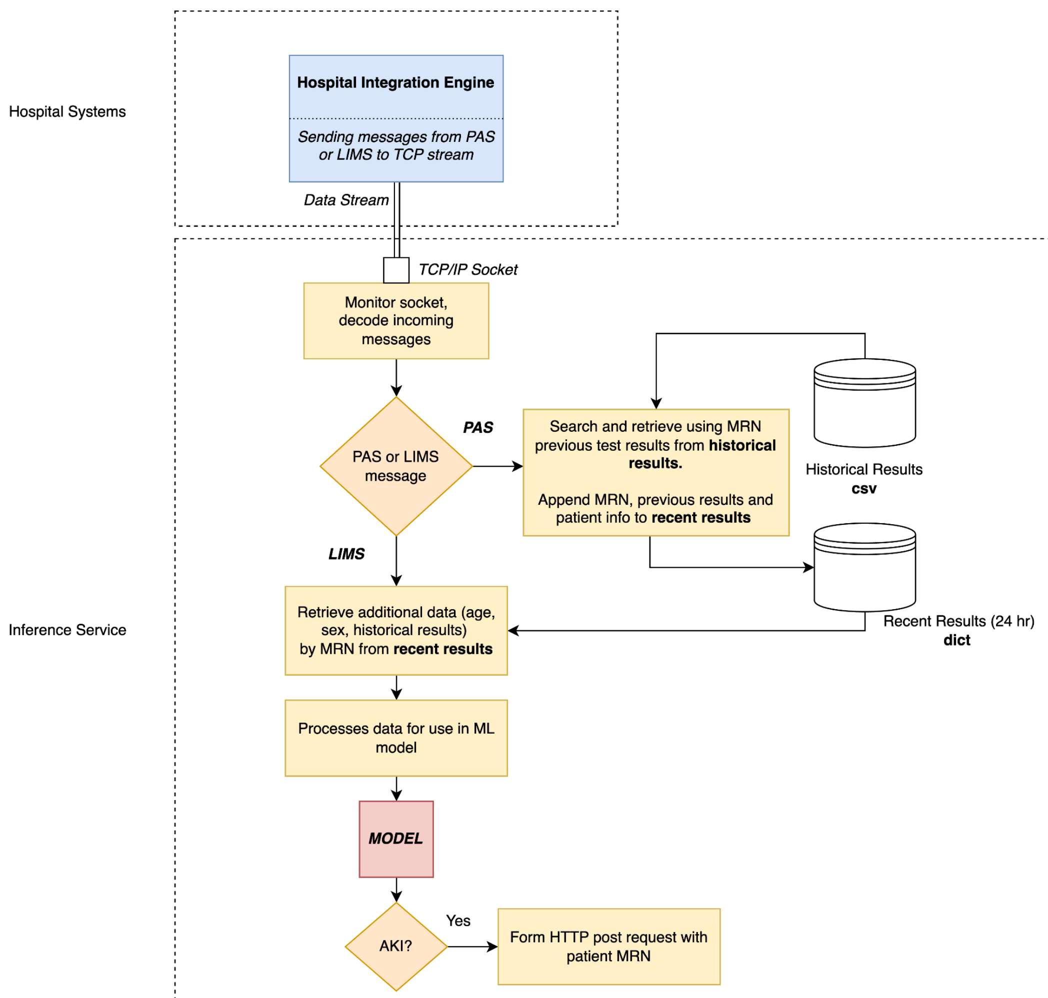
- Storing the output predictions from the AKI model
- Alerting the next of kin

Outline Solution

The outline of the proposed system will:

1. Listen on a TCP socket for the hospital's PAS and LIMS via the HL7 protocol
2. Parse the HL7 stream into semantic message types and preprocess for downstream storage in the database
3. When creatinine blood test result comes from the LIMS system, query the database to identify the patient's information and his/her historical results, using MRN, and run model inference to determine the presence of AKI (yes or no)
4. Make an HTTP post request to page the model prediction to the hospital's clinical response team.

Design Overview



Design details

There are four main components to our inference system: the **input processing layer** for decoding inputs from the hospital integration engine, **recent results dictionary** containing admitted patient information and their recent results, **preprocessor** for preprocessing data to input to the model and the **HTTP requester** to post requests to the hospital pager system.

The primary input to the inference system will be the datastream from the TCP/IP socket connected to the hospital's integration engine. The input processing layer decodes and extracts individual HL7 messages based on the MLLP protocol. Incoming HL7 messages are assumed to be either a PAS system message (new patient admission) or a LIMS system message (lab result).

If the incoming message is from the PAS system, a new entry in the recent results dictionary is created containing the patient information and hashed using the unique MRN. The historical results record is also queried using the MRN to find any previous test results to include in the dictionary entry. This structure is used so that when a new test result comes in patient data can be rapidly retrieved and used to inference with a model to obtain a result as fast as possible.

If the message is from the LIMS system, the included MRN is used to query the dictionary to obtain additional patient information. The LIMS result and additional information is preprocessed to be used with the model. If the model result indicates an AKI is likely then an HTTP post request to the pager system is immediately formed containing the patient's MRN.

Testing

In order to test the overall runtime of the system, the time module will be used to measure an average execution time across multiple sets of data. Edge cases of PMS systems will be implemented like empty messages, incorrect message types or disjointed data. We will also keep a withheld test dataset from the training to test the model on using scikit.metrics module to test the models f3 score. The system will be stress tested at 50 admissions and discharged with 100 blood test data to determine how it scales and whether it still performs in under 3 seconds.

Future work

The above document gives an overview of a proposed baseline product that can determine a binary result of AKI. However, via the NHS algorithm, there exists the potential to expand this project to reporting on the severity of the AKI and advising on the treatment. This can be incorporated into the model by collecting the training data, where the labels describe the severity of the AKI rather than a binary output. This can then be fed into our model to allow for classification but may come at a cost in execution time. Thus in order to potentially combat this, the database can pre classify those that are more at risk of AKI based on their historical data and already have a message ready to send if the blood tests confirm an AKI.

Updated Section

We made several key modifications to our initial plan to enhance the efficiency and reliability of our system. The first significant change was the integration of historical and new data into a unified database. This step involved converting the `history.csv` file into a dictionary format, using the patient's Medical Record Number (MRN) as the key, and their prior results as the values. When we received a PAS message, if the MRN wasn't already in our dictionary, we added a new entry with the patient's age and sex. For existing MRN entries, we simply updated the information with the latest age and sex data. This consolidation streamlined the process of retrieving a patient's previous blood test results during inference.

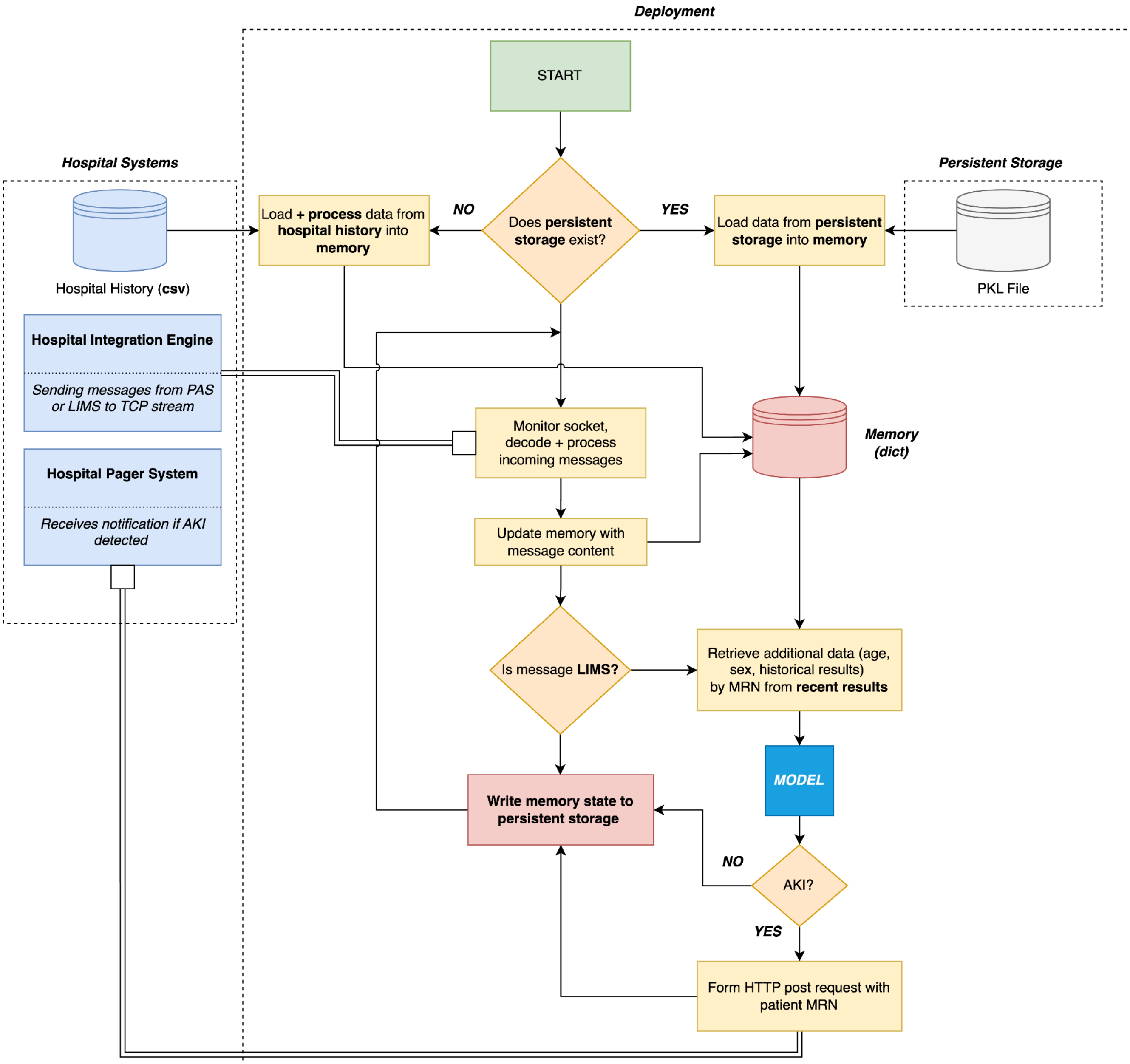
Another pivotal enhancement was the inclusion of the most recent blood test result in our results database. In our previous system, the blood test data from the LIMS message was utilized for inference but wasn't stored, meaning we lacked access to prior results for subsequent tests. By retaining the latest test result in our database, we significantly improved our model's predictive capabilities.

To address potential failure scenarios, we implemented the following proactive measures:

1. Persistent Storage of Results Database: Following the receipt of each LIMS or PAS message, we now save the entire database as a `.pkl` file in persistent storage. This approach guards against data loss in the event of a Kubernetes pod shutdown, ensuring data recovery even after a crash.
2. Resilient Connection to the Simulator: We introduced a while loop with a try-except clause for connecting to the simulator. This ensures continuous operation, enabling our code to automatically reconnect in case of connection resets.
3. Robust Connection to the Pager Host: Similar to the simulator connection, we implemented a while loop with a try-except clause for the pager host connection. This setup guarantees that pager requests can still be sent successfully after any connection failures.
4. SIGTERM Handling: To further safeguard data integrity, we added a SIGTERM handler that saves the results database into persistent storage whenever our code receives a SIGTERM signal. This feature ensures that crucial data is not lost if our code is instructed to terminate.

These enhancements not only improve the system's resilience and data consistency but also provide a more robust foundation for accurate and efficient patient data analysis.

Updated Design Overview



References

1. Wang, H. E., Muntner, P., Chertow, G. M., & Warnock, D. G. (2012). Acute kidney injury and mortality in hospitalized patients. *American Journal of Nephrology, 35*(4), 349-355. https://doi.org/10.1159/000337487
2. NHS England. (n.d.). Acute Kidney Injury Programme. Retrieved from https://www.england.nhs.uk/akiprogramme/