UNIVERSITY OF CALGARY

ENSF 609/610

AI Enhanced Soft Sensors

Midterm Report – Winter 2022

Group #11

Arman Sarraf, Behnaz Sheikhi, Chandrahas Reddy, Gabriel Mathias

Abstract

A chemical plant is an industrial process plant that manufactures (or otherwise processes) chemicals, usually on a large scale. The general objective of a chemical plant is to create new material wealth via the chemical or biological transformation or separation of materials. This process usually is done with sensors that measure the inputs and results important for process control. Soft sensors and hard sensors have their pros and cons. Since hard sensors are costly and sometimes hard to replace or fix, software sensors have been developed to make the process easier and more controllable. On the other hand, using soft sensors requires some initial costs to buy the license and have the knowledge and ability to work with. In this project, we try to establish a workflow, design, and build an application that connects the mathematical computation like the mentioned soft sensors and uses artificial intelligence and machine learning models to understand the logic behind calculating the results. The estimators then can be replaced with the machine learning models and used for the soft sensors to reduce the computation time and costs at the industry level.

Introduction

Process control of a chemical plant is guided by the many sensors found throughout the plant. These sensors can measure physical properties such as pressure, temperature, flow rate, etc. Data is usually transferred and logged into a database. Based on sensor readings, various actors such as operators, operation managers, plant managers, etc., make short-term and long-term decisions. For example, an operator might want to open a valve if a pressure sensor reaches a certain value to avoid dangerous pressure build-up in a vessel, or a reliability manager might want to study the effect of various parameters on the operation of the plant etc. Sensors cannot directly measure certain properties. However, these properties can be estimated using simple or complex mathematical algorithms using a software simulator with the model of the plant. The term soft sensor is applied to a sensor that collects signals from other sensors as input to a software model and outputs a processed signal. For example, the presence of ice particles flowing through a gas pipeline is not something that a sensor can measure. But, by knowing the composition, temperature, pressure, and flow rate of the gas, it can be estimated. Thus, a soft sensor could detect the presence of ice particles flowing through a gas pipeline to guide operators or develop maintenance schedules.

If used appropriately, soft sensors could allow for less hardware or resources on-site and lower capital and operating costs through optimization of data gathering. Another important feature of a soft sensor is that it can be predictive and use the exact mathematical correlations or AI, predict the occurrence of an event in the future, and take corrective actions before the event takes place. Soft sensors can be a tool to generate more accurate KPIs and guide high level decisions. The main objective of this project is to develop a framework to allow the creation of soft sensors. That is, to allow users to connect real-world data to software models that create data which cannot be directly measured. The following are the main objectives of the application:

- Plant operating data will be sent to an appropriate application that calculates user-defined properties from input data.
- Estimated properties must be pushed to a defined data destination such as a database.

- A machine learning model will be trained with the input data and results from the estimator.

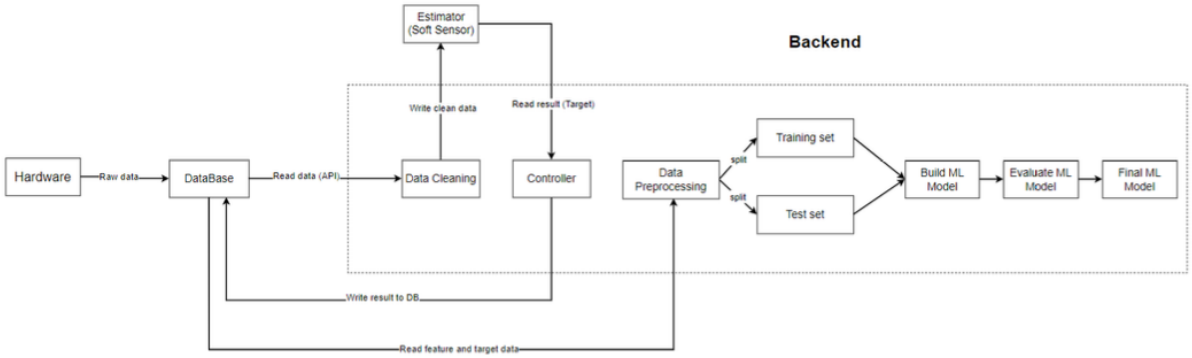

Figure 1: Conceptual diagram

Background

To develop a framework that allows the creation of soft sensors, various different topics had to be researched by the group. Topics ranged from understanding the problem at hand, such as what is a soft sensor, to more software engineering challenges such as how to get notified when a database receives a new entry. For all the mentioned challenges, the group had to research and become familiarized with to ensure the success of the project. The following is a list of some of the topics studies thus far:

- How hard and soft sensors work [1]
- What are MQTT protocol and Push/Subscribe mechanism [2]
- What is the estimator in the measurement process [3]
- How to connect the database to backend in Python [4]
- How to populate the database non-stop with intervals [5]
- How to implement Abstract, Interface, design patterns (observer, factory, singleton) in Python [6]
- How time-series machine learning models work [7]

Methodology

Challenges:

Several challenges have been identified throughout the project. The list of challenges identified while doing this project is as below:

- Getting plant data for testing purposes.
- Go from customer requirement to understanding workflow and development.
- Concepts of abstraction & now to design and implement in Python.
- Not having the domain knowledge of how soft/hard sensors work, and chemical background.
- Limited experience with time-series data & ML models for regression tasks.

- Not having access to customer real-world data.
- Making the database to get populated continuously.
- Use the proper approach to clean and preprocess the data to extract useful information from the real world's original data, including noisy data.
- Tuning the hyper-parameters to get the best score when a machine learning model is applied.
- Designing the application with enough flexibility can be easily overloaded and used by different soft sensors, databases, etc.

Solution:

In this project, Python programming language and related existing libraries were used to develop the backend. Initially, a pipeline diagram (Figure 1) was drawn to indicate how the entire system works and give a better understanding of the project's core. In the next step, design a required UML diagram (Figure 2) and implement design patterns for the classes to follow the solid principles of software design which are as follows:

Singleton pattern to SoftSensorManager class where that restricts the instantiation of a class to one "single" instance. Singleton Pattern is useful when exactly one object is needed to coordinate actions across the system.

The Strategy pattern was employed to enhance flexibility in the design of the backend. The pattern is employed by defining interfaces that must be implemented for each desired behavior. For example, there could be multiple implementations of the Estimator interface, each one representing a different strategy to estimate results.

Adapter pattern is implemented in the Estimator, DataStorage and MLModel classes. The pattern is required so that the soft sensor code is able to communicate with existing processes for estimators and data storages. The Adapter pattern is employed in the MLModel class so that the user is able to use a machine learning model without having to code the model himself.

Observer pattern implemented for DataStorage and SoftSensorManager class to perform events to drive the software. SoftSensorFactory class creates a SoftSensor object after getting the type of the sensor the customer needs to apply on their data.

To be able to test the code, a database had to be designed in such a way that new data would come in periodically. This way, we can test whether the soft sensor can recognize the addition of new data and run the appropriate estimator to fetch the results. We implemented two call procedures called InsertPE for inserting values into the table and LoadPE for where to load the values by using InsertPE procedure. The database is populated with an interval (8 seconds) to model the incoming data.

Implemented a GUI called tkinter package which is a standard python interface to get database credentials from the customer where it contains login details and user selected inputs, outputs and what kind of soft sensor customer needs where mysql.connector.connect is used to connect the backend to the database. Based on customer input the backend starts the workflow and results the output. An LSTM-Model is developed where it trained on a train set of the incoming data and

provided predictions for the test set. For evaluating the performance of the ML model the RMSE (Root Mean Squared Error) metric is used and this ML model can be used as an AI estimator.

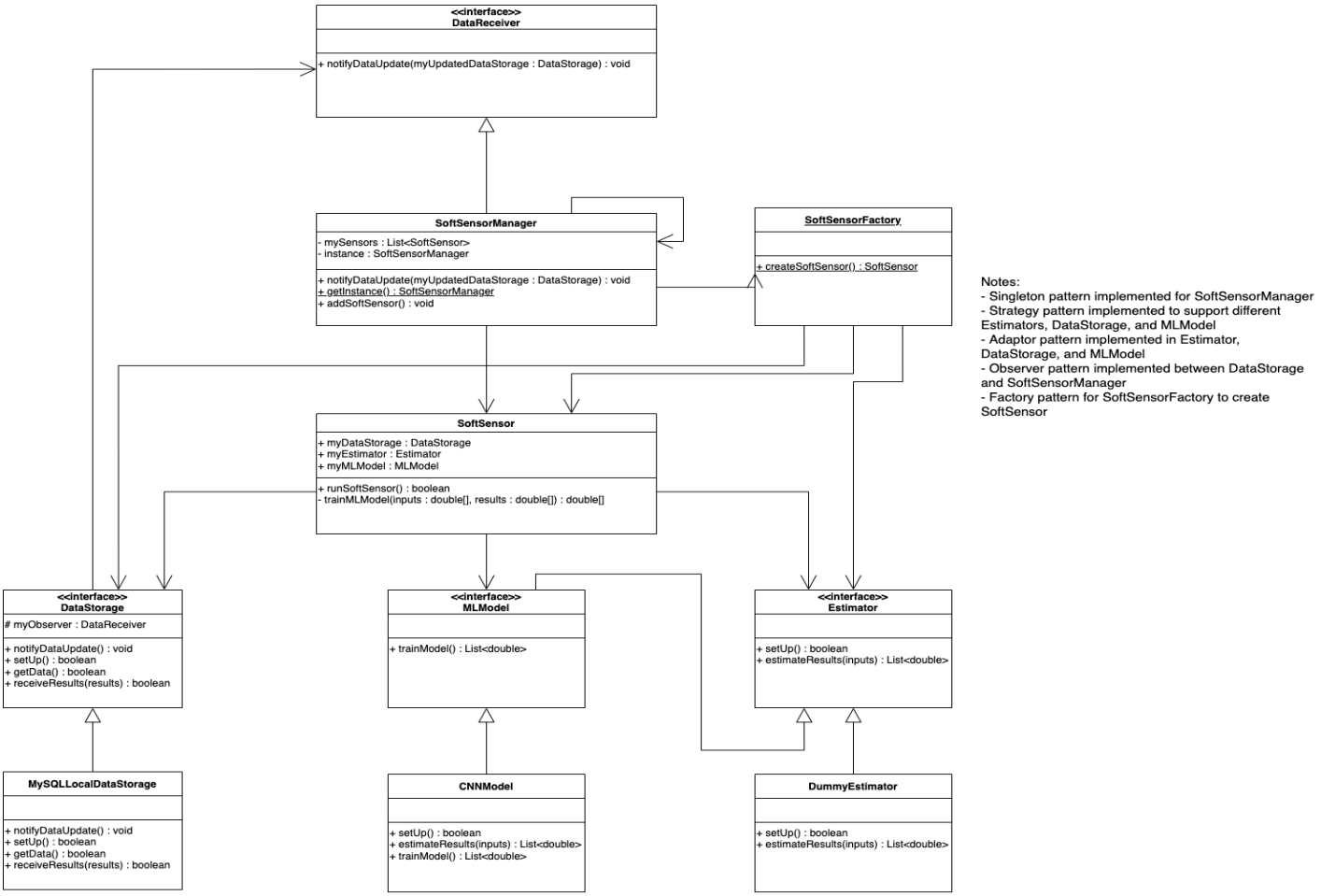

Figure 2: UML Diagram

Evaluation:

To evaluate the process that has been done during this step of the project, we can consider the project progress and the performance of the algorithms used in the project.

Project Progress Performance

To address the project progress performance, the tasks completed are as follows:

- The pipeline of the workflow and required UML design. Based on the UML required classes including Abstract, Interface by use of inheritance and design patterns.
- Database to connect to the backend to get query, update, and write data created. It is populated in a fixed interval to model the incoming data.

- A soft sensor estimator is implemented to estimate the value of the output column based on the input columns. Input and output columns are selected by the customer through the implementation of a GUI.
- A LSTM-Model was designed and trained by incoming data that can be used as the AI estimator.

To evaluate the progress and performance of the project during development, the ongoing evaluation at regular intervals (weekly) were used to monitor development time and the quality of the feature development to make sure it met the planned expectations. Based on the introduced deadlines and the scope in the project proposal in this phase, all the parts were covered before their due date.

Algorithm Performance & Benchmark

The algorithms implemented in this project are done through a soft sensor estimator. The soft sensor gets a list of the new incoming data which includes the records of selected input columns and estimates the value for the output column throughout some calculations on the input values. Since this calculation must be done on the list of records, the minimum time complexity would be O(n) which n is the number of records in the list, and the maximum time complexity depends on the calculation strategy. For example, in an implemented estimator we just calculate a value for the output column based on one input column, so the time complexity is O(n).

Test Plan/Results

In order to test the existing workflow two separate classes such as “DataGenerator” and “Application” were developed. The first class is used to generate the data constantly and fill the database table in a fixed interval (8 seconds). While the second class is implemented to test the process of the designed pipeline started by receiving user inputs (through GUI) such as database credentials to create the new database connection based on the selected database and user information, input columns to read from the database, and output columns to be calculated through estimator. Then a soft sensor will be created. After that, the existence of new data in the fixed interval will be checked (pull mechanism), if the new data is available, it will send the records to the estimator to estimate the value of selected output based on the input columns. In the next step, the database table will be updated with the estimated value for output. In the final step, the data will be passed to the model to do preprocessing, training on the train set, prediction on validation data, and then calculate the model performance using some metrics. The result of this process is explained in the Machine Learning Results section.

Machine Learning Results

A deep-learning based model named LSTM-Model also trained on the incoming data and provided predictions on the test set. In order to evaluate the performance of the ML model the RMSE (Root Mean Squared Error) metric is used in a regression task, it calculates the accuracy and performance of predicted value. A perfect RMSE value is considered 0.0, which means that all predictions matched the expected values exactly. In our case, after training and evaluating the model, we reached 0.03 RMSE which could be considered promising and near the desired target. Figure 3 shows the training and validation improvement after training the model.

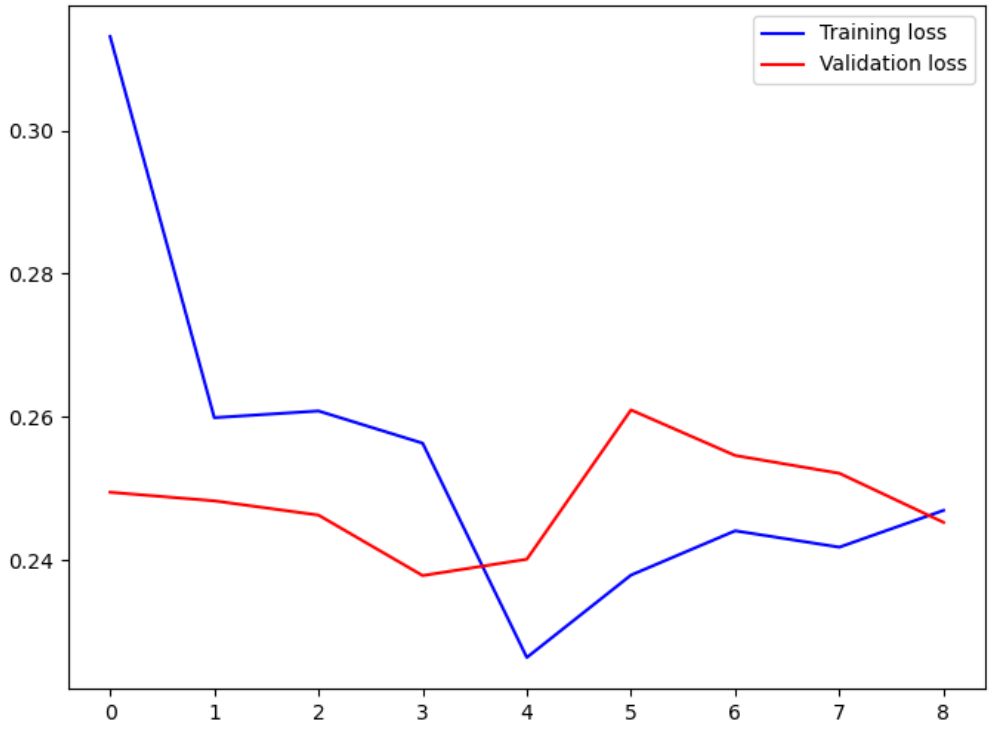

Figure 3: Plot of the LSTM training process

Recommendations going forward

In this scope of the project, we implemented an estimator with a method of calculation, and used it to provide estimation for the output column. To add more flexibility to the system we can add different types of estimators with a different approach to calculate the value for the outputs. Furthermore, we create a soft sensor in the backend and test the workflow of the project. In terms of more flexibility, we can receive the soft sensor type through the GUI and create it based on the customer data.

Plans going forward

A project proposal with a tentative timeline was submitted in January 2022. In the table below, all items identified in the project proposal are present along with a status update. The figure below is a Gantt chart of how the project will progress to completion. The original scope of the project mentioned other areas the project could expand into if time allowed. Some items were finished ahead of time which allowed the opportunity to expand the project scope. The new items in the project scope are also present in the table below.

Table 1: Project deliverables updates.

Description	**Original Due Date**	**New Due Date**	**Status Update**
Research available APIs to connect to databases	February 28th	Done	Different APIs were researched for database connectivity. The SQL database was chosen for the purposes of this project
Develop infrastructure to connect to a database and read data	February 28th	Done	Connectivity to an SQL database was implemented and tested on a locally hosted database.
Develop infrastructure to write to the database	February 28th	Done	Connectivity to an SQL database was implemented and tested on a locally hosted database.
Generalize infrastructure to allow for any schema	April 11th	March 18th	System is flexible at this point. Interfaces have been created to establish contracts for communication between the project and the processes it has to communicate with. There is an opportunity to generalize the data structure used to pass information between the methods.
Create communication between a software model and database (the user will give this software model)	April 11th	April 11th	Database still to be supplied by the user.
Create a deep learning model that is trained on the background based on database inputs and software model outputs	April 11th	Done	The ML model is trained every time the soft sensor is run. Model is saved so that it can later be loaded.
Create a GUI	If time allows	March 25th	A simple GUI has already been implemented. Opportunity to make the GUI look nicer.
Implement other databases' technologies to be supported by the backend	If time allows	April 11th	A pull strategy has been implemented where a soft sensor constantly checks whether a database has new data. A push strategy is to be developed where the soft sensor is notified of a data update the time it happens.
Implement other estimator technologies to be supported by the backend	If time allows	April 11th	Implementation of a ML learning estimator that loads a ML model to be used. The team will investigate a self training estimator. Look into the implementation of an Excel sheet estimator.

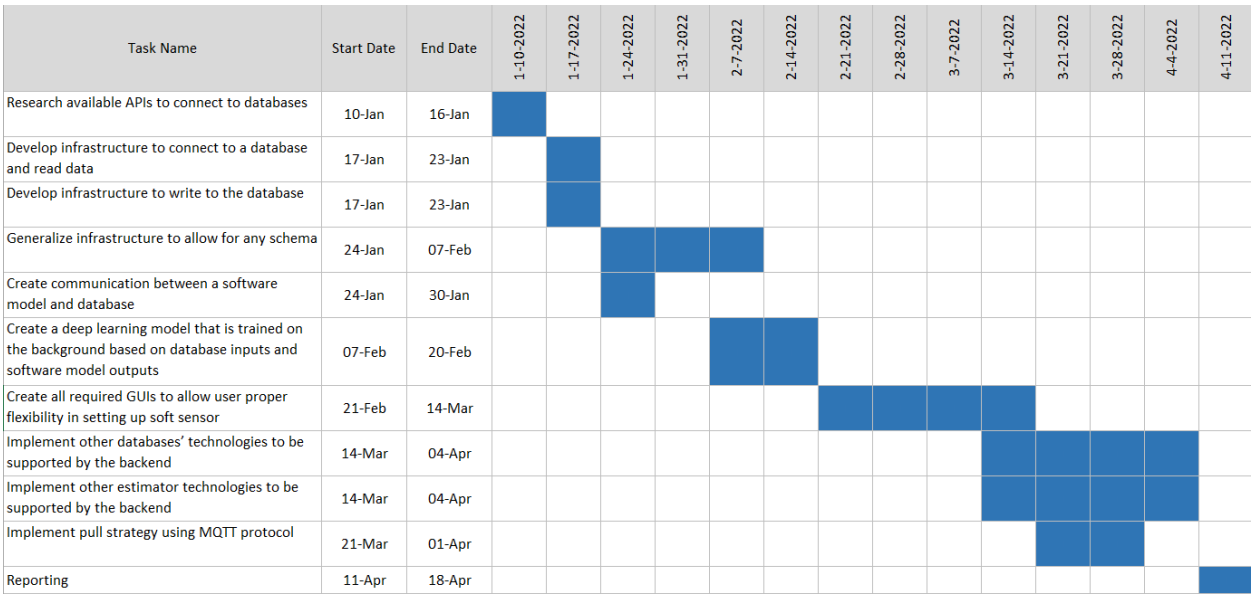

Figure 4: Gantt chart of project deliverables

At the end of this project, we will deliver a framework to develop soft sensors for real world data. The soft sensor will be able to connect to a SQL database where new data is input. Whenever new data comes in, the soft sensor will run the estimator to get results which will be saved to the database. In the background, a ML model will be trained and saved. The user will be able to choose to load this ML model as the new estimator.

Reflections and conclusions

Regarding what has been mentioned so far, the first step was to understand the customer needs, what must be delivered as a final application, and what important problem our solution will solve. Then, after translating the requirements to the proper pipeline, we designed the necessary diagrams and system workflow to have a deep understanding of the required abstract classes and interfaces and implement them with concrete classes. As a result, the implementation includes creating a SQL database, the estimators which receive the raw plant data and return some estimated output, a pull mechanism for fetching new coming data into the backend, a deep-learning-based AI model named LSTM, and some user interfaces to communicate with the customer. On the other hand, it is observed that the flexibility of the application should be increased to cover different types of estimators and have better communication with the user, which will be considered in the next steps.

From the project management point of view, one of the most profound lessons learned is what the customer requires. The team members learned that understanding what the customer wants to have at the end of the project is the most important part of any project. In addition, there have been several useful lessons learned from the technical aspect so far. The most important ones could be considered as follows. Getting familiar with plant data and understanding how hard/soft sensors work in the pipeline. Learning how to implement OOP in the python programming language, using interfaces, and connecting them to concrete classes, how design patterns work, how to deal with numeric data, and the understanding of implementing memory-based models such as LSTM and

understand what metrics we should use to calculate the performance of the trained model to make sure it is trained successfully.

After evaluating and checking the project progress with the supervisor and presenting the progress to the class in the midterm presentation, we found that we were on the right track for most parts of our plan. On the other hand, regarding the deadlines set at the beginning of the project, we observed that we were on top of our schedule. However, it is noticed that while we go deeper into the implementation parts of the application, some specific details are necessary to be added to the next step implementations. Also, some new criteria should be added to the existing workflow, which we were unaware of at the beginning of getting familiar with the concept and the entire project.

References

[1] https://www.sciencedirect.com/science/article/abs/pii/S0098135409000076?via%3Dihub

[2] https://mqtt.org/

[3] Discussions with our supervisor

[4] https://pynative.com/python-mysql-database-connection/

[5] Push vs. pull strategy was discussed with the supervisor

[6] https://www.tutorialspoint.com/python_design_patterns/python_design_patterns_singleton.htm

[7] https://machinelearningmastery.com/how-to-develop-lstm-models-for-time-series-forecasting/