# Assignment 2

Victor Arvidsson
*DV2586 Deep Machine Learning*
*Blekinge Tekniska Högskola*
Karlskrona, Sweden
viar19@student.bth.se

## I. Introduction

The goal in this assignment is to use an autoencoder with LSTM cells in order to detect anomalies in a sensor data stream. This assignment will use the CRSIP-DM approach to development, excluding the first and last steps of the process.

### A. CRSIP-DM

The CRISP-DM approach is a data mining process for data analysis and data mining [1]. This can be also be applied to machine learning problems, since the most important part of any machine learning project is data management. The process consists of six parts: Business understanding, Data understanding, Data preparation, Modeling, Evaluation, and Deployment. These steps are visualized in Fig. 1. For this assignment, we do not have any business understanding, since we do not have any stakeholders to discuss with. We will also skip the deployment step, since that is beyond the scope of this assignment.

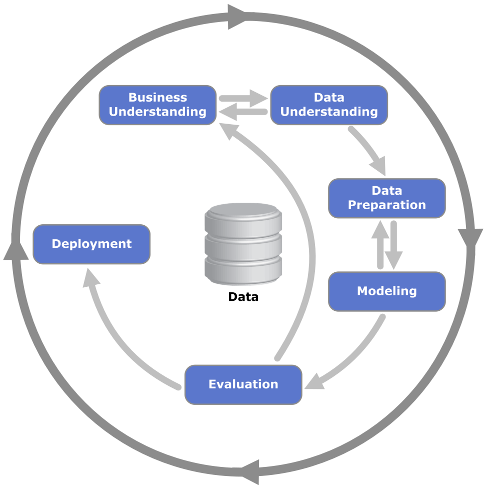

Fig. 1. A visual overview of the CRISP-DM process.

## II. Data Understanding

The data set provided in the assignment was used for this task. It contains randomly generated values for three different types of sensor data. These are temperature, pressure, and humidity. For each data point there is also a label that specified if the data point is within normal values, or if it should be regarded as an anomaly. We also have a time stamp for each data point. There are 1000 data points in total in the data set, with values measured every 5 minutes, for a resulting time span of almost 3 days and 12 hours. A plot of the data can be seen in Fig. 2. For the sake of simplicity, we will focus on the temperature for the remainder of the assignment, but the same principles should apply equally well to the other two attributes. A plot of the temperature data can be seen in Fig. 3. Plotting the temperature with the anomalies in a different color, as can be seen in Fig. 4, we can see that there is no correlation between the data points in the time series. We can also see a very clear line of distinction between the normal and anomalous data points.

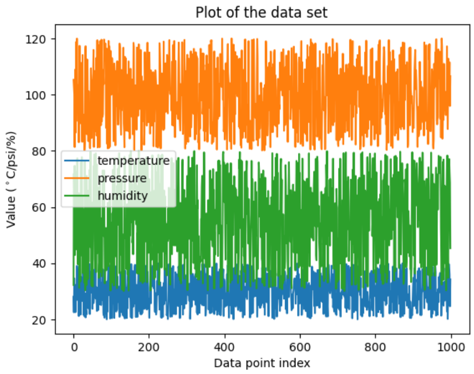

Fig. 2. Plot of the entire data set.

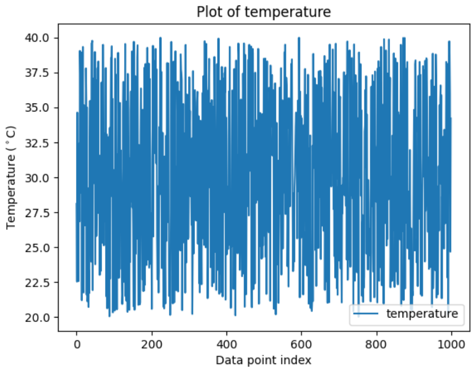

Fig. 3. Plot of the temperature values.

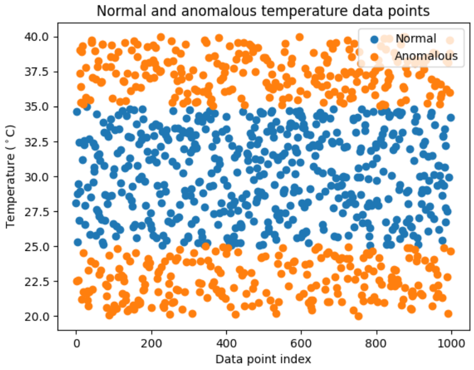

Fig. 4. Plot of the temperature values with anomalies in a different color.

## III. Data preparation

For the data preparation, since we don't see any distinct outliers or missing values, we don't need to do any data cleaning, and instead directly start with data munging. We begin by dividing the data set into two parts, one training data set and one testing data set. We do this immediately to reduce the problem with data leakage. We then use min-max normalization to rescale the values to be between 0 and 1. One important thing to note is that the min and max values are taken from the training data set and then used for both the training and testing data sets to avoid data leakage.

From the training data set, we create a new data set that only contains the normal data points, since these are the ones that we will use to train the autoencoder.

For the test data set, we divide it into three parts, one with all of the normal data, one with the anomalous data that is larger than the normal data, and one with the anomalous data that is smaller than the normal data.

Finally, in order to use the auto encoder model, the data sets needs to be divided into shorter signals. Through testing, it was determined that a window size of 5 worked well. In order to maximize the number of signals that the model could train on, and the number of signals that we could use to evaluate the model, we use a sliding window technique to create the signals, with a window size of 5 and a stride of 1.

## IV. Modeling

For the model, we use an autoencoder with LSTM cells. The autoencoder consists of an encoder and decoder part. The encoder part down-samples the input in order to create an encoded representation that can then be up-sampled to recreate the original signal. For this application, we use symmetrical encoder and decoder networks. This simplifies the process of finding an optimal network structure. For all of the layers we use ReLU as the activation function for the cells, and sigmoid as the recurrent activation function.

There are two layers in the encoder and two layers in the decoder. The first layer of the encoder has 8 units, while the second layer has 4. For the second layer we only output the last state to get a 4x1 representation of the 5x1 input vector.

For the decoder part of the network, we reverse the layer sizes. We also start with repeating the input vector so that we have a full length signal to send to the decoder. To receive the final output for the network from the last layer, we consolidate the output from the LSTM units for each of the time steps using a dense layer with one output node.

An overview of the model structure, but with slightly different parameters, can be seen in Fig. 5.

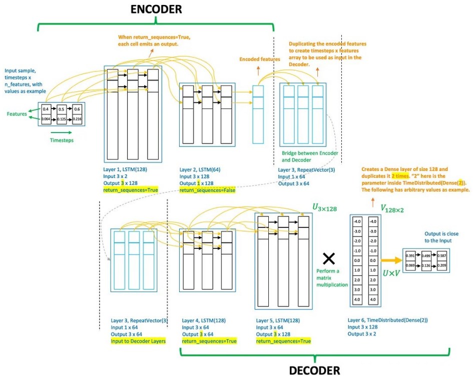

Fig. 5. Overview of a similar model structure, but with 2 features, a window size of 3, and larger LSTM layers [2].

In order to use the autoencoder for anomaly detection, we use the reconstruction error. By only training the model on normal data, we expect that the reconstruction error will be higher for anomalous data compared to normal data. Therefore, by using the mean and standard deviation for the reconstruction error for the model, we can define a threshold for the error. If a signal is reconstructed with an error above the threshold, we classify it as an anomaly, otherwise we regard it as normal.

## V. Evaluation

For the model, we use Adam as optimizer and mean squared error (MSE) as the loss function. To evaluate the model performance, we use mean absolute error (MAE). The model is trained for 100 epochs. The training data set consists of 407 signals and the testing data set consists of 185 signals, of which 91 are considered normal. The resulting training and validation loss can be seen in Fig. 6, and the training and validation MAE can be seen in Fig. 7.

After training, the mean MAE for the training data set is 0.130, and the standard deviation is 0.073. We define the threshold as being one standard deviation above the mean, which results in a threshold reconstruction error of 0.203. Using this threshold, and defining anomaly as the positive class, we get the following results:

| | |
|---|---|
| TP | 94 |
| TN | 85 |
| FP | 6 |
| FN | 0 |
| Accuracy | 96.76% |
| Precision | 94.00% |
| Recall | 100.00% |
| F1-score | 96.91% |

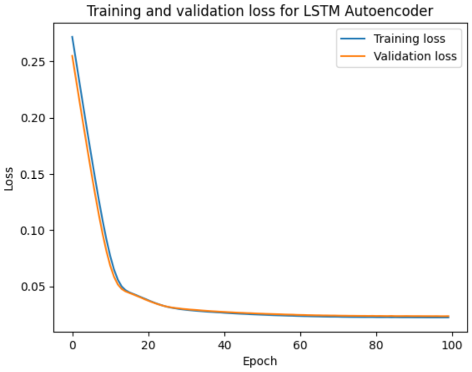

Fig. 6. Training and validation loss.

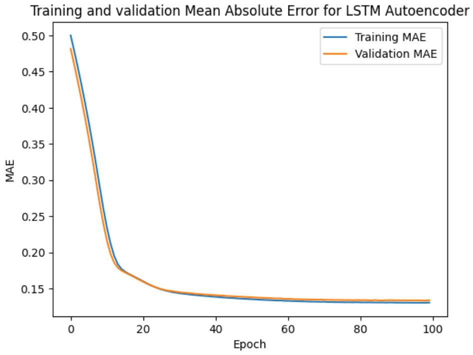

Fig. 7. Training and validation MAE.

## VI. Discussion

As we can see from the results, we manage to achieve a high accuracy and F1-score. Since the bottle neck in the autoencoder only has 4 cells, while the input signal has 5 time steps, we can also conclude that we can successfully compress a signal and recreate it from it's compression. We also see no over- or underfitting when looking at the loss curves.

The method used in this assignment to divide the data into one normal data set and one anomaly data set worked well for this problem. The reason for this is that there is no correlation between the data points in the time domain, and such, the timestep does not provide any information to the model. We can therefore disregard the timeseries component of the data and rearrange the data points as we want without losing any information. For a real world data set that is gathered from real sensors, we would expect there to be a correlation between the data points in the time domain, i.e. if the temperature was 26°C 5 minutes ago, we would expect it to be close to 26°C now as well, we would not expect it to jump directly to 33°C within that time frame. In that case, we would need another strategy to divide the dataset into normal and anomalous signals. However, in such a scenario, we would also expect to find longer periods of normal data, making it easier to create continuous signals of normal data. In the provided data set, the longest continuous normal signal has 9 values, and there are 13 normal signals with a length of 5 or longer, providing us with a maximum of 25 signals that we could use for training or testing. Therefore, not ignoring the time series component would leave us with far too few training signals to use for our model.

In section II, we reasoned that the model should work equally well for the other two attributes as well. Using the same model as we used for temperature, we get an accuracy of 98.92% and F1-score of 98.80% for the pressure attribute, and an accuracy of 94.05% and an F1-score of 93.25% for the humidity. This confirms our reasoning from section II.

## References

[1] M. S. Brown, “What IT Needs To Know About The Data Mining Process,” Jul. 2015. [Online]. Available: https://www.forbes.com/sites/metabrown/2015/07/29/what-it-needs-to-know-about-the-data-mining-process/

[2] C. Ranjan, “Step-by-step understanding LSTM Autoencoder layers,” Feb. 2022. [Online]. Available: https://towardsdatascience.com/step-by-step-understanding-lstm-autoencoder-layers-ffab055b6352