CONVOLUTIONAL NEURAL NETWORK FRAMEWORK FOR ASSET MANAGEMENT

PATTERN RECOGNITION OF SENSOR DATA TIME-SEQUENCES

Luis H. PINTO — e-mail: luishenriquepinto.73@gmail.com

February, 2020

Abstract

Condition-monitoring (CM) plays an important role in maintenance strategy and it is becoming the central contributor to asset health status indicators and early detection of failures. Small, low-cost sensors can measure a vast number of physical quantities such as vibration-levels, energy-consumption, temperature-gradients, electrical-currents, and in the most part of the cases, unusual variations in the patterns of these quantities precede failure events. Together, the massive amount of data generated by these sensors and **Machine Learning (ML)** technique can now capture and learn the operational patterns of the assets, creating models that perform the evaluation of their health status.

Currently, a large number of ML techniques are being tested, resulting in different values in performance and accuracy. **Convolutional Neural Network (CNN)** is an ML technique generally used for image recognition due to the ability to recognize complex patterns from time-series of sensor data. This article uses a CNN framework, more precisely a **One-Dimensional 1D-CNN** to evaluate health status of assets from sensor data resulting in very high accuracy (approximately 99%), certainly above the average capacity of the most common ML techniques for the same type of use.

1. Introduction

Machine Learning (ML) has become one of the cornerstones of asset management based on component conditions, but countless experiences show that the challenges behind ML are not easy. ML solutions for predicting health status are very heterogeneous due to the large number of failure modes to be mapped and predicted. The main objective of ML solutions for **Condition-based Maintenance (CbM)** is to find a fairly simple way to recognize the pattern deviation associated with component malfunction, and for this situation depending on the level of importance of the component to the system, the accuracy of the response generated by the ML solution becomes a differential factor.

The idea in this article is to use a framework derived from **One-Dimensional Convolutional Neural Network (1D-CNN)** in a CbM problem. 1D-CNN is a ML solution commonly used for **Image Recognition (IM)** and **Natural Language Processing (NLP)** problems. Its high capacity to identify and and derive complex features from shorter (fixed-length) segments of the overall data set becomes an interesting ability to be used in condition asset management.

The CbM problem in this case is based on experimental data from a hydraulic module which consists of a primary working and a secondary cooling-filtration circuit connected via the oil tank. The module repeats constant load cycles (duration 60 seconds) while the temperatures and cooler efficiency measurements are recorded. The cooler is the component under evaluation and its performance is attested according to the heat-exchange capacity ranging from 100% (as good as new condition) to $< 3\%$ (total failure condition). Four temperature sensors are installed in specific points of the hydraulic circuit in order to pass all the necessary data to the ML model. This configuration ensure that the 1D-CNN is able to derive the maximum quantity of features during the functioning of the physical system.

The data can be found in the **GitHub** link Condition Monitoring Hydraulic Data Set, including a detailed description depicted in the README.md file.

2. Data Description

A brief summary of the data set is presented in the Tab. 1 with a total of 531,405 values. Each sensor is recorded 2,205 times during 60 seconds, and the cooling efficiency is attested at the end of each measurement cycle according to three categorical status (i.e.: 100%, $< 20\%$ and $< 3\%$).

Sensor	**Physical quantity**	**Unit**	**Sampling Rate**	**Instances**	**Attributes**
TS_1	Temperature	$^\circ\mathrm{C}$	1 Hz	$2,205 \times 60$	Time-serie, real
TS_2	Temperature	$^\circ\mathrm{C}$	1 Hz	$2,205 \times 60$	Time-serie, real
TS_3	Temperature	$^\circ\mathrm{C}$	1 Hz	$2,205 \times 60$	Time-serie, real
TS_4	Temperature	$^\circ\mathrm{C}$	1 Hz	$2,205 \times 60$	Time-serie, real
CE	Cooling efficiency	%	1 Hz	2,205	Categorical, real

Table 1. Summary of the data set.

A small sample of the temperature profile recorded by the TS_1 and TS_2 sensors is shown in the Fig. 1. The physical model is not an important concern for solving this problem; therefore, no emphasis will be given on the behavior of temperatures during the operating cycle of the hydraulic module.

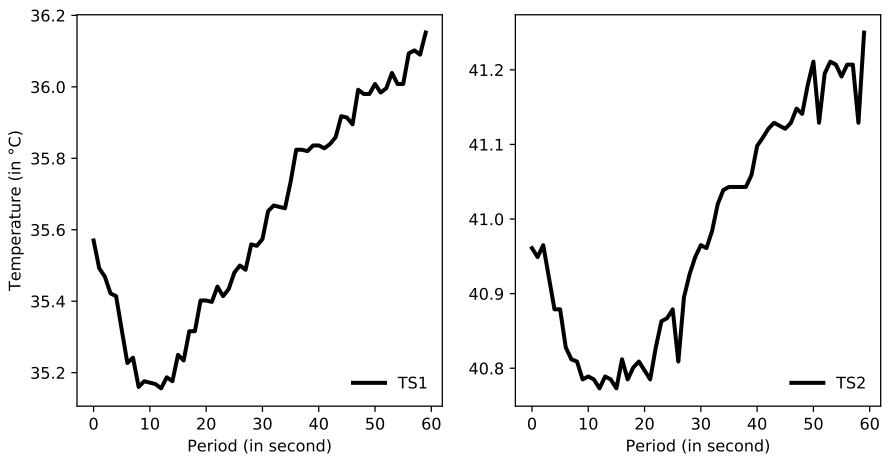

Figure 1. Temperature profiles recorded by the TS_1 and TS_2 sensors — in $^\circ\mathrm{C}$.

3. The 1D-CNN Model

As previously mentioned, a 1D-CNN is a class of **deep neural networks**, a regularized versions of a **Multilayer Perceptrons (MLP)**. MLP usually mean fully connected networks, that is, each neuron in one layer is connected to all neurons in the next layer. The "fully-connectedness" of these networks makes them prone to overfitting data. Typical ways of regularization include adding some form of magnitude measurement of weights to the loss function. 1D-CNN take a different approach towards regularization: they take advantage of the hierarchical pattern in data and assemble more complex patterns using smaller and simpler patterns. Therefore, on the scale of connectedness and complexity, 1D-CNN are on the lower extreme.

The name Convolutional Neural Network indicates that the network employs a mathematical operation called **convolution** which is a specialized kind of linear operation.

CNN are simply neural networks that use convolution in place of general matrix multiplication in at least one of their layers.

The design of a 1D-CNN consists of an input and an output layer, as well as multiple hidden layers. The hidden layers of a 1D-CNN typically consist of a series of convolutional layers that convolve with a multiplication or other dot product. The activation function is commonly a **RELU** layer, and is subsequently followed by additional convolutions such as pooling layers, fully connected layers and normalization layers, referred to as hidden layers because their inputs and outputs are masked by the activation function and final convolution.

In order to respond to the CbM problem depicted in the Sec. 1 a 1D-CNN is proposed according to the Fig. 2.

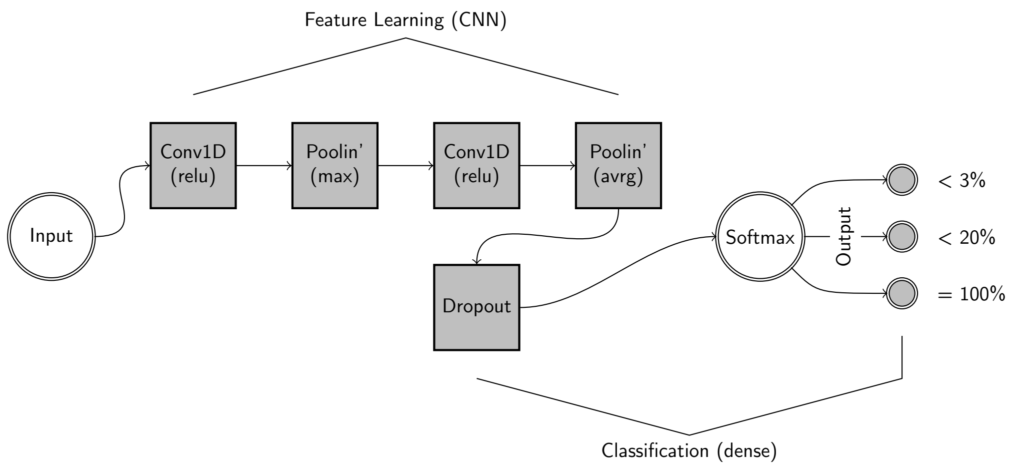

Figure 2. One-Dimensional Convolutional Neural Network (1D-CNN)

with the following characteristics:

- $1 \times$ **input** layer with dimensions $(2{,}205 \times 60 \times 4)$ corresponding to the number of time-series in the data set (2,205), the length of the time-series (60) and the quantity of mapped sensors (4);
- $2 \times$ **1D-Convolutional** layers with dimensions (50×100), 160 feature filters and kernel-size equal to 6, activation function RELU;
- $1 \times$ **Max Pooling** layer with dimensions (16×100), patch equal to 3;
- $2 \times$ **1D-Convolutional** layers with dimensions (50×100), 160 feature filters and kernel-size equal to 6, activation RELU;

- $1 \times$ **Average Pooling** layer with dimension (16×100), patch equal to 3;
- $1 \times$ **Dropout** layer; and
- $1 \times$ **Dense** layer with dimension 3, activation function SOFTMAX, corresponding to the outputs (i.e.: efficiency $< 3\%$, $< 20\%$ and $= 100\%$);

In this configuration the network is divided in two zones: the **Feature Learning** zone corresponding to the convolution/polling layers where the data features are mapped in order to enlarge and better differentiate the data set; and the **Classification** zone, a conventional MLP process used to classify the data series coming from the convolution zone.

The 1D-CNN model shown in Fig. 2 can be assembled by using **Keras Python Deep Learning** library through the following program lines:

```
mdl = Sequential()
mdl.add(Conv1D(100,6,activation = 'relu',input_shape = (60,4)))
mdl.add(Conv1D(100,6,activation = 'relu'))
mdl.add(MaxPooling1D(3))
mdl.add(Conv1D(160,6,activation = 'relu'))
mdl.add(Conv1D(160,6,activation = 'relu'))
mdl.add(GlobalAveragePooling1D())
mdl.add(Dropout(0.5))
mdl.add(Dense(3,activation = 'softmax'))
```

A summary of the entire 1D-CNN after the compilation of the lines above mentioned is illustrated in the Tab. 2.

ID	Layer	Output shape	Parameters
Conv1D-1	One-Dimensional Convolutional	(55, 100)	2,500
Conv1D-2	One-Dimensional Convolutional	(50, 100)	60,100
Poolin'-1	Max Pooling	(16, 100)	0
Conv1D-3	One-Dimensional Convolutional	(11, 160)	96,160
Conv1D-4	One-Dimensional Convolutional	(6, 160)	153,760
Poolin'-2	Global Average Pooling	(160)	0
Dropout-1	Dropout (drop parameter = 0.5)	(160)	0
Dense-1	Dense MLP	(3)	483

Table 2. 1D-CNN summary.

The total number of parameters is equal to 313,003 all them corresponding to weights and bias of the neural network.

4. Simulation & Conclusions

The complete evaluation of the 1D-CNN was performed by using 80% of the data set for training, and 20% for testing. The evolution of the **accuracy** and the **loss** of the framework is shown in the Fig. 3.

Short explanation about accuracy and loss: The accuracy of a model is usually determined after the model parameters are learned and fixed and no learning is taking place. The loss is always calculated on training and validation and its interpretation is how well the model is doing for these two sets. The higher the accuracy, the better a model, and in the opposite sense, the lower the loss, the better a model (unless the model has over-fitted to the training data). According to the Fig. 3 both indicators, accuracy and loss, show that the 1D-CNN designed for the CbM problem is well succeed in determining the status of the cooler from its temperatures time-series.

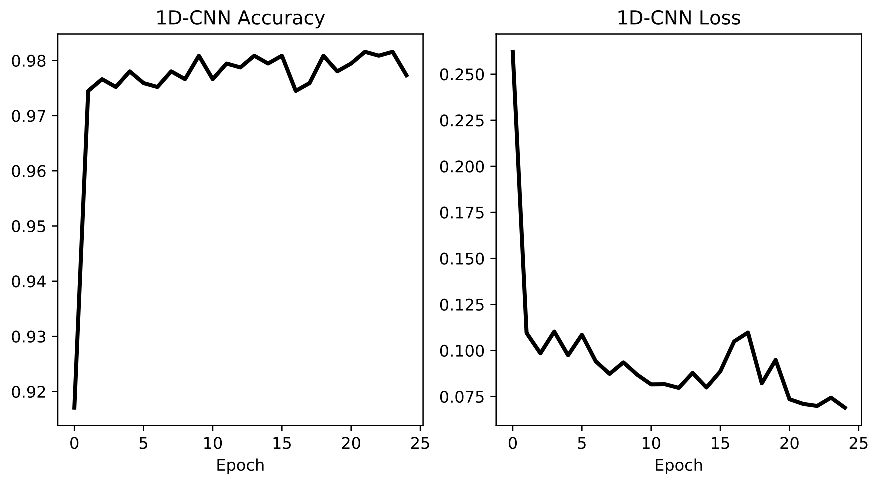

Figure 3. Evolution of the 1D-CNN accuracy and loss according to the training epoch.

One more indicator used to qualify the 1D-CNN in terms of precision is the **confusion matrix** illustrated in the Fig. 4. The matrix shows a high precision (> 98%) in all the three situations required by the problem, which is in fact a very effective solution to classify time-series according to the class or status of the component/equipment, in case of a CbM problem.

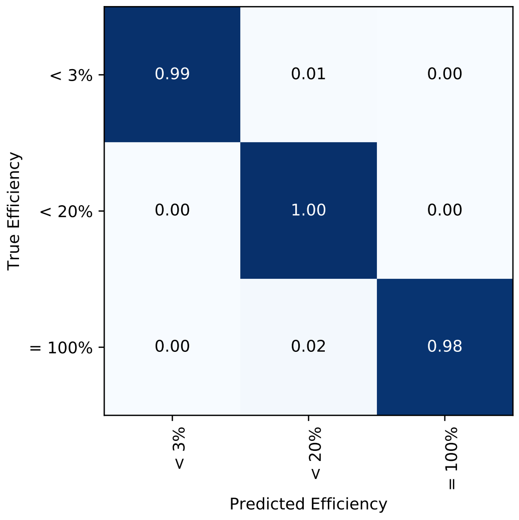

Figure 4. 1D-CNN confusion matrix.

Formally, the 1D-CNN proves to be a powerful tool for anticipating or predicting failures based on field-data recorded by sensors, since they can expand the amount of data by using as many feature filters as the user wants, which in fact creates substantial differences from small variations in the data.

The computational effort is relatively low, since the convolution zone (see Fig. 2) in the network performs simple mathematical functions during the evaluation of the data set.

A complete version of the Python files used in the simulations can be found in the **GitHub** link Condition Monitoring Hydraulic Data Set, including a detailed description depicted in the README.md file.

5. About the Author

Luis H. Pinto is Mechanical Engineer, specialized in Automotive Engineering, Business Management, and Thermal Sciences, 21 years experienced as technical manager with emphasis on optimization of manufacturing processes of automotive industries, researching and development of methodological methods for qualitative and quantitative analysis of equipment and process reliability, including the use of deep learning concepts.

luishenriquepinto.73@gmail.com

References

[1] Marco Cerliani. *Predictive Maintenance: Detect Faults from Sensors with CNN*. 2019.

[2] Marco Cerliani. *Remaining Life Estimation with Keras*. 2019.

[3] François Chollet et al. *Keras: The Python Deep Learning Library*. 2015.

[4] Simon Haykin. "Feed-forward Neural Networks: An Introduction". In: (2004), pp. 1–16.

[5] Renu Khandelwal. *Convolutional Neural Network(CNN) Simplified*. 2018.

[6] Willamos Silva et Miriam Captrez. "CNN-PDM: A Convolutional Neural Network Framework For Assets Predictive Maintenance". In: (2019).

[7] Luis Pinto. "Data-driven Method for the Prediction of Equipment Remaining Useful Life — Clustering and Neural Network Pattern-Recognition". In: (2019).

This document was written in LaTeX format.