

On-Edge Implemented Machine-Learning Based Synthetic Flame Detector For Gas Turbine Operation

Valentina Gori^{1,*}, Kanika Goyal², Tiziano Roma¹, Gianni Bagni¹, Riccardo Carta¹, Bruno Giunta¹ and Giovanni Tonno¹

¹Nuovo Pignone Tecnologie s.r.l., a Baker Hughes company, via F. Matteucci 2, 50127 Firenze (Italy)

²Baker Hughes, Kundalahalli Colony, Brookefield, 560037 Karnataka, Bangalore (India)

Abstract

We developed a synthetic flame detector based on a 55-parameter feed-forward neural network receiving as input seven physical quantities measured at different locations of a gas turbine. The model has been implemented on an edge device for the machine control (MarkVIe) and the synthetic flame detection has been performed in real time. The model has achieved full recall and full precision on an independent test set. Further improvements will be aimed at implementing the model on a MarkVIIs device, so that the control loop can be closed.

Keywords

Synthetic flame detector, virtual sensor, digital twin, control, edge deployment, gas turbine

1. Introduction

When operating a gas turbine, flame-out detection is a crucial safety-related check to be put in place. In fact, in case of an accidental flame-out in the combustor, fuel gas can accumulate in the gas turbine and an explosion may occur, if no prompt action is taken to stop the fuel gas injection.

Physical devices for flame detection are available on the market. Still, Artificial Intelligence (AI) technologies, such as a synthetic flame detector, may improve the detection performance or completely substitute the physical sensor, with a significant cost reduction.

We show how we trained a data-driven synthetic (or virtual) flame detector and implemented it on MarkVIe.

AI4DT&CP@IJCAI2024: The Second Workshop on AI for Digital Twins and Cyber-Physical Applications in conjunction with 33rd International Joint Conference on Artificial Intelligence, 3rd-9th August 2024, Jeju, South Korea

*Corresponding author.


✉ valentina.gori@bakerhughes.com (V. Gori); kanika.goyal@bakerhughes.com (K. Goyal);

tiziano.roma@bakerhughes.com (T. Roma); gianni.bagni@bakerhughes.com (G. Bagni);

riccardo.carta@bakerhughes.com (R. Carta); bruno.giunta@bakerhughes.com (B. Giunta);

giovanni.tonno@bakerhughes.com (G. Tonno)

 0000-0003-0215-4022 (V. Gori)

 © 2024 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

2. Method

2.1. The dataset

The dataset used for training the model was acquired from a gas turbine prototype running on a test bench. Temperature, pressure, flow, speed, acceleration values were measured by probes installed on different sections of the machine and acquired with a sampling frequency of 1 Sample/40 ms. With the same sampling frequency, the continuous (s) and boolean (b) signal from a physical flame detector installed on the turbine were also acquired. The boolean b , whose value is calculated by the acquisition system based on the value of s , and has value 1 whenever there is flame, 0 whenever the flame is out. In the training dataset the proper behaviour of the physical flame detector installed has been assessed by subject matter experts. A total amount of about 760 kSamples, corresponding to 7 days of gas turbine operation, have been used as training and validation sets (with an exclusive splitting with ratio 70% vs 30%), while an independent dataset counting about 1.1 MSamples and acquired during 10 test days has been used as our test.

2.2. The model

We leveraged a fully data-driven approach and trained a simple fully-connected neural network (NN) [1] with 55 parameters and one hidden layer to learn the behavior of the flame. We trained a regression model with a supervised approach, using the continuous signal s coming from the physical flame detector as our ground truth. Basically, we built a digital twin or virtual sensor [2] of the physical flame detector. The chosen input features were seven and include speed, flow, pressure, and temperature measurements sampled at different sections of the turbine.

The ground truth used to train the model is the continuous signal s coming from the physical flame detector, and the model is a regression model which outputs \hat{s} . A boolean signal \hat{b} has been calculated out of the model prediction \hat{s} as well, to give a binary information to the control system. Also, this allows to compute recall and precision metrics of our model more easily. The logic used to discretize \hat{s} is:

$$\hat{b} = 1 \text{ if } \hat{s} > thr \quad (1a)$$

$$\hat{b} = 0 \text{ otherwise} \quad (1b)$$

where the threshold thr is such that recall and precision (see Sec. 2.3) are maximized on a validation set, where the model boolean output is compared with the physical detector boolean output acquired.

The choice of the input features and the NN hyperparameters was led by the maximization of the recall and precision (see Sec. 2.3) on an independent validation dataset. Also, we preferred architectures which would keep the number of parameters as small as possible, so that this solution could be easily implemented on MarkVIE, which is our edge device.

2.3. The metrics

Recall and precision are computed on an event basis. We have a true positive when these conditions are satisfied:

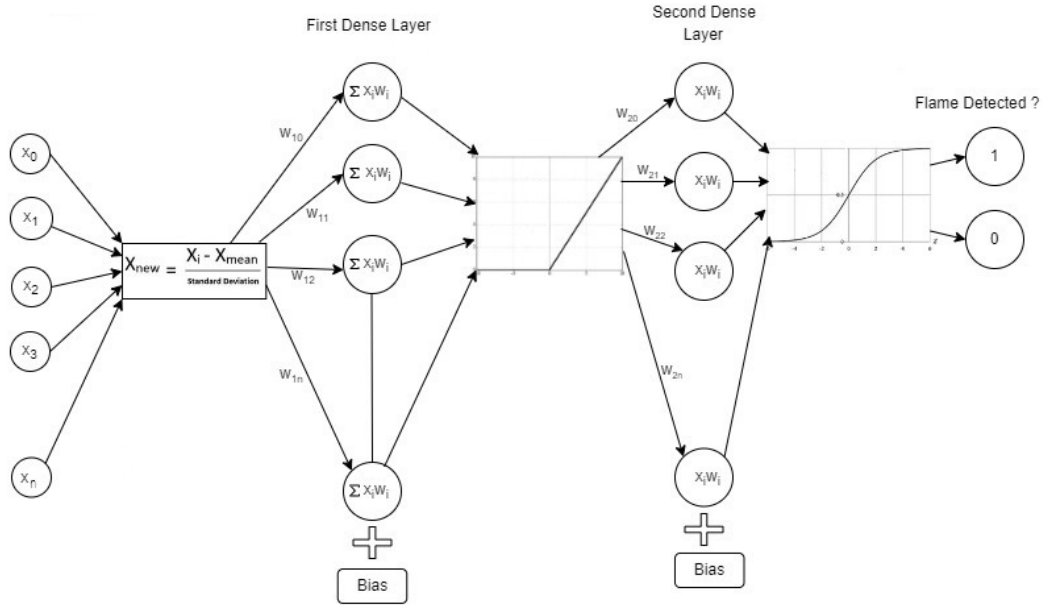


Figure 1: MarkVie architecture: each block represents an elemental operation: standard scaling of input data, matrix multiplication, matrix addition, ReLu for non-linearity and sigmoid for final prediction.

1. b switches from 1 to 0 at a certain time instant t_{out}
2. \hat{b} switches from 1 to 0 at time $t_{out}^{\hat{}}$, such that $|t_{out} - t_{out}^{\hat{}}| < \Delta t$

The time tolerance Δt is 1.2 s, corresponding to 30 samples, and derives from functional requirements. In fact, we do not want to have a late flame out detection, since this may cause explosion risks due to methane accumulation. We may thin to relax this constraint to 40-50 samples in case of an earlier detection, which instead is a desired behaviour.

Analogously with the definition of a true positive, a false positive is given when:

1. \hat{b} switches from 1 to 0 at time $t_{out}^{\hat{}}$
2. b keeps on having value 1 in the time frame $[t_{out}^{\hat{}} - \Delta t; t_{out}^{\hat{}} + \Delta t]$

Finally, we have a false negative when:

1. b switches from 1 to 0 at time t_{out}
2. \hat{b} keeps on having value 1 in the time frame $[t_{out} - \Delta t; t_{out} + \Delta t]$

2.4. The deployment on edge

MarkVie [3] is a flexible controller for multiple applications. It features highspeed networked I/O for simplex, dual and triple redundant systems. Industry standards Ethernet communications along with USB and COM ports with 667 MHz Processor and QNX operating system are used for I/O and supervisory interface to operator and maintenance stations.

Many techniques were explored to deploy flame detector model onto MarkVie Edge Controller:

1. Converting model weights into ONNX format and further generate xml of functional blocks that could run on MarkVIe
2. Converting model into a Dynamic Link Library
3. Developing the NN model using the coding language of MarkVIe

We used the third approach. The necessary steps are detailed as follow:

- Convert the trained model into equations whose coefficients are the model weights
- Map the equations onto MarkVIe functional blocks
- Create the functional block library to represent the activations of the neural layers
- Deploy and test the implementation on a virtual controller
- Implement it onto production controller
- Check that the inference time is compatible with the requirements of real-time application.

As shown in Fig. 1, the model deployed on MarkVIe consists of one block for each elemental operation. The architecture starts with the scaling of sensor data read from the gas turbine, followed by the multiplication by the weights of the first dense layer. The output is further added with the bias and a ReLu is applied to introduce non-linearity. The weights of the second layer are multiplied by the output from ReLu, then the bias is added and finally a sigmoid operation is performed to have the final output.

3. Results

The model was producing real-time information about the flame status (on/off), given that the inference time is less than 40 ms, which is the sampling interval used by the control system. We tested the model on the same gas turbine from which training data were collected. The test set was chronologically after the training set.

We obtained full recall, meaning that all the flame-out events were correctly detected, and full precision, i.e. no false alert for spurious flame-out was given by the model, as shown in Fig. 2. The flame-out was always detected slightly in advance with respect to the physical flame detector (also see Fig. 3), allowing a larger safety margin.

The model performance were checked both real-time, while the gas turbine was running, and offline, for a deeper analysis of results, by downloading data from MarkVIe.

4. Conclusions

This work shows a real-case application of a synthetic flame detector consisting of a small neural network implemented on a MarkVIe edge-device and producing real-time information on the control system of a gas turbine.

The synthetic flame detector is trained using temperature, pressure, speed, flow time series collected by sensors installed on the machine, and leveraging the time series acquired from a physical flame detector as our ground truth. Basically, a digital twin of the physical flame detector sensor is built.

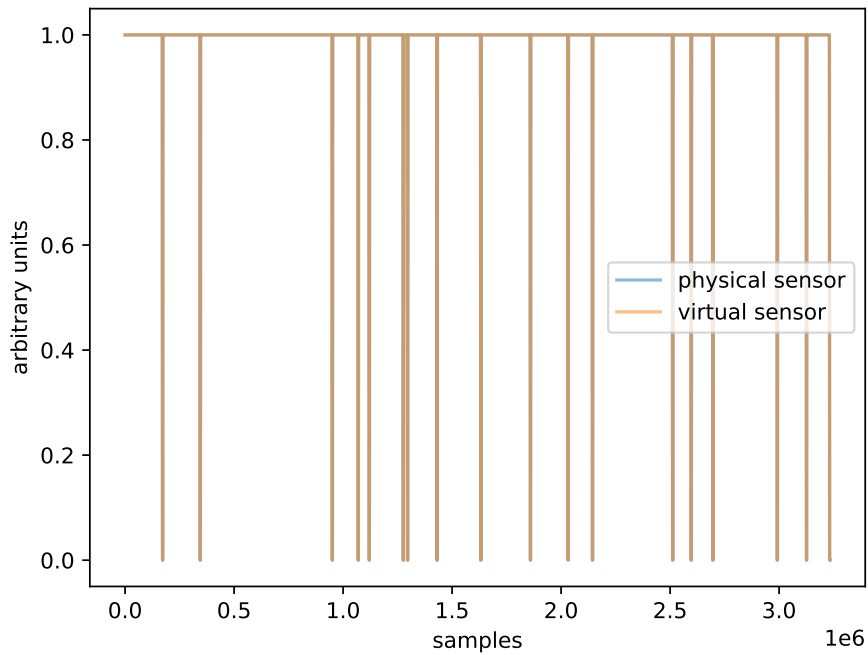


Figure 2: Comparison between the boolean output from the physical sensor (in blue) and from the virtual sensor (in orange), on a dataset collected during two months operation, for a total amount of about 3.5 MSamples. The boolean signal has value "1" when the flame is on, "0" when the flame is off.

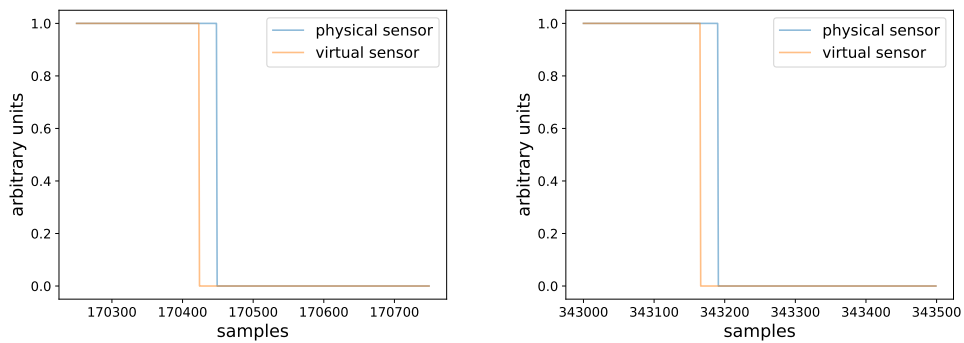


Figure 3: Detail of two flame-out events. The virtual sensor (orange line) detects flame-out slightly in advance with respect to the physical sensor (blue line).

The model allows a full recall and full precision in the flame out detection on a test set independent from training and validation sets. The deployment on MarkVIe allows to have the information about the flame status in real time, while the gas turbine is running, so that control engineers are able to suddenly shut down the machine in case of a flame out, thus preventing

an explosive environment to be generated.

The next step will be to use this flame detector in the control closed loop. To achieve this, we will need to deploy the model to MarkVIs, which is an enhanced version of MarkVIe aimed at automatically manage security-related machine controls.

References

- [1] I. Goodfellow, Y. Bengio, A. Courville, Deep learning, MIT press, 2016.
- [2] D. Martin, N. Kühl, G. Satzger, Virtual sensors, Business Information Systems Engineering 63 (2021). doi:10.1007/s12599-021-00689-w.
- [3] <https://www.governova.com/gas-power/products/digital-and-controls/mark-vie-ecosystem>, Accessed: 13 May 2024.