PROCEEDINGS OF SCIENCE

Data Center IT Anomaly Prediction and Classification: an INFN CNAF experience

Luca Torzi,[a] **Elisabetta Ronchieri,**[a,b,*] **Luigi Benedetto Scarponi,**[a] **Luca Giommi**[a] **and Alessandro Costantini**[a]

[a] *INFN CNAF*
Viale Berti Pichat 6/2, Bologna, Italy
[b] *Department of Statistical Sciences, University of Bologna,*
Via Belle Arti, 41, Bologna, Italy
E-mail: luca.torzi37@gmail.com, elisabetta.ronchieri@cnaf.infn.it, luigi.scarponi@cnaf.infn.it, luca.giommi@cnaf.infn.it, alessandro.costantini@cnaf.infn.it

The INFN CNAF data center provides a huge amount of heterogeneous data through the adoption of dedicated monitoring systems. Having to provide a 24/7 availability, it has started to assess artificial intelligence solutions to detect anomalies aimed to predict possible failures.
In this study, the main goal is to define an artificial intelligence framework able to classify and predict anomalies in time series data obtained from different sensors and systems within the data center (i.e., electrical plant, cooling system, and UPS system). Having to deal with unlabeled data, the proposed framework performs as a first step a regression task to learn the behavior of the sensors and, given the previous 5 timestamps, provides the values of the sensors in the next timestamp. As a second step, it performs a classification task. Comparing the predicted and the actual behaviors of the sensors, in fact, evaluates the status of the system and possible anomalies. During the first step, a mean squared error of 0.025 has been obtained, while in the second one an F1-score of 0.997 has been reached.

International Symposium on Grids and Clouds (ISGC2024)
24 -29 March, 2024
Academia Sinica Computing Centre (ASGC), Institute of Physics, Academia Sinica Taipei, Taiwan

*Speaker

PoS(ISGC2024)006

1. Introduction

Data center IT hosts sensors that can control and monitor, among others, the state of the chiller plant, the electrical plant, and UPS. It is recommended to detect anomalies (i.e., events deviating from what is considered to be standard, normal, or expected) to keep the data center IT running. Anomalies can arise from various factors, including reaching critical chiller temperatures and sensor inaccuracies.

The INFN CNAF data center provides data for several components that are 24/7 controlled and monitored by experts. For the electrical plant, the cooling system, and the UPS system, we have already performed a study that detects anomalies by using three different approaches [1]: one is based on the mean and the standard deviation of the sensors values, well known as Z-score; another one is based on the percentiles; and the third one leverages Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise (DBSCAN) [2]. By using these solutions, we have grouped and identified anomalous events.

In this work, the proposed framework performs firstly a regression task to learn the behavior of the sensors and, given the previous 5 timestamps, provides the values of the sensors in the next timestamp. Secondly, it performs a classification task and finally, by comparing the predicted and the actual behaviors of the sensors, it evaluates the status of the system and possible anomalies.

The following open-source python libraries have been used to perform this study: Pandas [3] to read the csv files and process the dataframes; Matplotlib [4], Seaborn [5] and Plotly [6] to plot figures; SciPy [7] to compute statistical measurements over the data; scikit-learn [8][9] to preprocess the data and to implement machine learning techniques.

2. Sensors Data Analysis

The majority of the collected data have a time window that begins on January 6, 2022, with the exception of the UPS system data, which starts on May 31, 2022. Some data that sum together the values of different sensors (like the file that contains the sum of the power consumption of the refrigerator units) have started to be collected on May 17, 2023 and have less than 5k values. Whereas, other files have more than 50k values. All the gathered data time windows ends on July 7, 2023.

Most of the sensors values are sampled every 15 minutes, however, some sensors (like the UPS system) are sampled every 10 minutes. Merging together more sensors data with the timestamp as key requires to insert a tolerance of 15 minutes in order to obtain rows where every timestamp have values for each sensor.

To detect any statistical relationship, whether causal or not, between two variables of our data we have computed the Pearson's correlation coefficient: it is obtained by taking the ratio of the covariance of the two variables in our numerical dataset, normalized to the square root of their variances. Mathematically, one divides the covariance of the two variables by the product of their standard deviations [10]. To compute it over a dataframe, a method of scikit-learn has been used that returns a matrix containing the correlation values computed between pairs of sensors: the value is 1 if the variables are directly linear correlated; 0 if the variables are not correlated; -1 if the variables are indirectly linear correlated.

2.1 Cooling system data exploration

The cooling system contains some data that come from sensors measuring the temperature of the incoming and outgoing water to the refrigerator unit and the evaporator temperature, which have values ten times bigger than the real values. In this case, values have been scaled by a factor 10.

Furthermore, the cooling system data are divided in seven blocks related to six different refrigerator units (RU) and other global measurements. Each refrigerator unit contains 7 sensors that measure: the power consumption in kilowatt (kW); the pressure of the cell 1 in bar; the pressure of the cell 2 in bar; the environment temperature in °C; the evaporator temperature in °C; the incoming water temperature in °C; the outgoing water temperature in °C.

The sensor that measures the outgoing temperature of the refrigerator unit 5 contains some errors in the measurements until mid-May 2022. The measurements that exhibit these errors are around 6350. Therefore, we have excluded it from the analysis, consequently reducing the number of entries for this refrigerator.

In every correlation matrix of the refrigerator units, except the RU2, a semi-strong correlation between the power consumption and the pressure of the cells of the unit itself is observed. Furthermore, in most of the units there is an high correlation between the environment temperature and the pressure of the cells.

The other files contain sensors data that measure pumps 1-2 power consumption in kW; pumps 3-4 power consumption in kW; environmental temperature of the refrigerator units; outgoing water temperature 1 in °C; incoming water temperature 1 in °C; outgoing water temperature 2 in °C; incoming water temperature 2 in °C; total power consumption of the refrigerator units in kW; total power consumption of the pumps in kW. The last two files have been omitted in the analysis because they have less than 5000 rows. The correlation between the delivery temperature and the return temperature of the two pumps is really strong.

Between refrigerator units there is respectively a semi-strong correlation between the power consumption of the units 1, 2, 3, 5 and the power consumption of the units 4, 6.

Instead the power consumption of the pumps 1-2 and 3-4 do not seem correlated to the other measurements, but they have a similar trend.

The system sends alarms if the outgoing temperature overcomes 15 °C or if the incoming temperature overcomes 20 °C.

2.2 Electrical plant data exploration

This system contains three transformers, so the sensors data are divided in three blocks. Each sensor in the transformer measures: the current phase 1 in A; the current phase 2 in A; the current phase 3 in A; the neutral current in A; the cosine of phi; the apparent power in voltampere (VA). There is one more file measuring the total apparent power of the transformers, that begins the measurement on May 18, 2023, and contains less than 5000 rows.

Two transformers are always in operation, while the third is in reserve. In the time window considered, the transformer 1 is always working. The transformers 2 and 3 take turn: from March 23, 2022, the transformer 3 stopped working, and the transformer 2 started to operate.

In every transformer, we have noticed a strong correlation greater than 0.97 between the current phases 1, 2, 3 and the apparent power. Moreover, in the transformers 2 and 3, we have also observed a strong correlation greater than 0.97 between the neutral current and the cosine of phi.

2.3 UPS system data exploration

The data of the UPS system are measured from only one sensor, so there is one single file. The sensor measurements are sampled every ten minutes and the recording begins on May 31, 2022. Furthermore, the data in the UPS system starts with some delay with respect to the others and this can bring a reduction of the size of the dataset, therefore they are omitted from the analysis. We have also neglected the UPS data due to they show a steady behaviour on the power consumption, equivalent to flat signals.

2.4 Comparison among sensors

To compare data of different sensors we have first considered the same time window from May 2023 up to July 2023. Then, we have focused on data from the cooling system and the electrical plant.

Figure 1 shows that, in the measured time window, there is a 0.71 correlation between the total power consumption (PC) of the refrigerator units (RUs) and the total apparent power of the transformers (TRs).

Figure 2 shows a strong correlation greater than 0.95 between both the incoming (i.e., delivery) and the outgoing (i.e., return) temperatures. Furthermore, there is also a middle-strong correlation, between 0.71 and 0.82, of these measurements with the total power consumption (PC) of the pumps. The trend of the total pumps PC is similar to the trends of the other temperature variables, where the position of the peaks is the same in all the plotted trends.

3. Dataset definition

The sensors used are related to the electrical plant and the chiller area with more than 50k samples, so that the files containing the sum of multiple sensors and that have more or less 5k rows are removed to avoid having a small dataset.

The values of each sensor are merged together to create a unique dataframe containing a timestamp and all the recorded values of the sensors. To merge them a Pandas API has been used, called *merge_asof* [11]: it merges the values of multiple dataframes based on the values of a row. In this case the time information is used, with a tolerance of 15 minutes.

To train a neural network, it is necessary to build a dataset containing anomalous and non-anomalous timestamps. To identify them, firstly we have examined the behavior of the sensors in a timestamp. We have also considered the anomalies detected by DBSCAN in the majority of the sensors, and the anomalies detected by the Z-score method for the sensors measuring the temperatures [1].

Once identified when a given entry in the sensor represents an anomaly, a classification method can be used to classify it with the first label if at least one sensor in the chiller and one sensor in the electrical plant behave in an anomalous way or if at least three sensors altogether behave in an anomalous way.

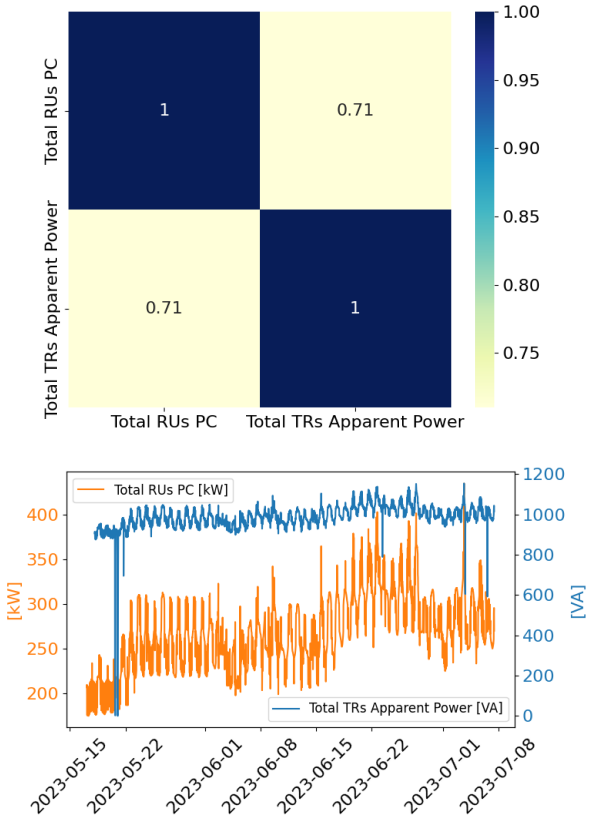

Figure 1: Correlation and trends of the total power consumption of the refrigerator units and the total apparent power of the transformers.

The dataframe contains 50568 rows as non-anomalous timestamps, and 1889 rows as anomalous timestamps. To train in a faster way the neural network, the dataframe has been normalized such that the values of the non-anomalous timestamps are in a range that goes from 0 to 1, then the anomalous timestamps are normalized according to the values of the first set (because the *MinMaxScaler* function [12] is fitted on the first dataframe).

4. Deep Learning approach to detect anomalies

Our deep learning solution implements two tasks into two different networks. The first one performs a regression task, therefore it learns the behavior of the sensors and, given the previous 5 timestamps, it returns the values of the sensors in the next timestamp. The second one performs a classification task, therefore, given the predicted and the actual behaviors of the sensors, it returns if the timestamp under consideration is anomalous or not.

4.1 Regression task

Xie et al. [13] describe an architecture that contains two main blocks characterized by a dynamic graph attention variant (GATv2) layer [14] and the Long-Short term memory (LSTM)

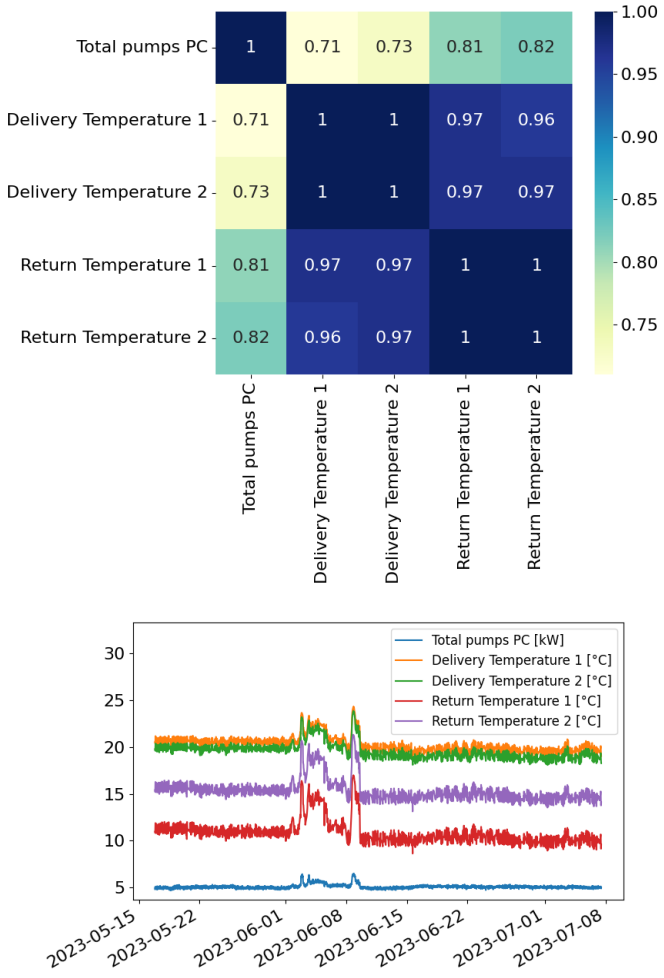

Figure 2: Correlation and trends of the total power consumption of the pumps and the incoming/outgoing temperatures.

layer [15]: one is able to capture the relations with other sensors (spatial operation); one is able to capture the trend of the sensor itself, considering the time series (temporal operation).

Before passing the values of the sliding window to this part of the network, we have defined an embedding in order to obtain a new representation of the sensors in a larger space. This is done through the usage of a Linear layer.

During the performing of the spatial operations, to avoid that information coming from different timestamps are merged together, n different GATv2 are used, one for each timestamp in the sliding window. For the same reason, when computing the temporal operation, 66 different LSTM layers are used, such that each one could be easily learn to the behavior of one single sensor.

At the end the activation of the LSTM layers is processed by a last Linear layer that will provide the results.

Ioffe and Szegedy [16] state that an important component of a neural network is the normalization in order to perform the training phase faster and less influenced from the initial random initialization of the parameters. So, we have performed a batch normalization after each GATv2

layer and each LSTM layer.

4.2 Classification task

Once the regression network has returned the expected behavior of the sensors, we have compared it with the observed one.

A simple neural network based on two Linear layers have been defined. After the first one there is a batch normalization and a ReLU. The network returns two numbers between 0 and 1, representing the probability that the timestamp is anomalous and the probability that it is not anomalous.

The first network has been trained using the Mean Squared Error as a loss function. For the second one the Cross Entropy is used as loss function.

The network is able to properly learn from this unbalanced dataset, the minority class is weighted 10 times more than the majority class.

5. Training Phase

To train the regression network we have considered the non-anomalous timestamps, instead to train the classifier we have used both.

To avoid holes between timestamps, we have not randomly sampled the dataset to create the train, validation and test sets. Therefore, we have not used the cross-validation procedure. Furthermore, it is better if the ratio between non-anomalous and anomalous timestamps is preserved in all the sets.

To achieve all these points, we have firstly divided the dataset in six different parts of equal length, then we have created the training set, the validation set and the test set of each part: they are not randomly sampled, but consecutive timestamp are selected. This number gives us the preservation of the ratio between the number of anomalous and non-anomalous data points in all the sets. Moreover, each set may also see different parts of the data points.

Figure 3 shows how the dataset is divided. The whole dataframe is divided in 6 equally long parts. Then each part is divided in three parts: the first 70% is for the training set, the second 10% is for the validation set and the last 20% is for the test set. Then these parts are merged with the respective parts of the other division to create the three sets.

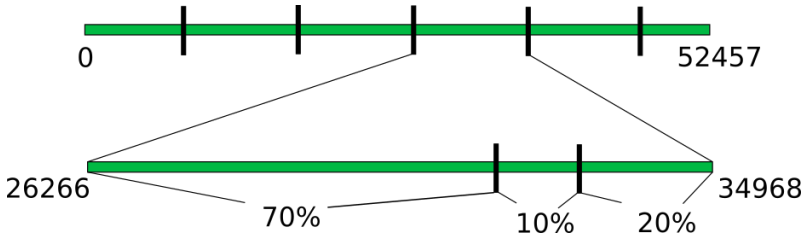

Figure 3: Dataset splitting.

Notice that this division in train set, validation set and test set is preserved in both the network, this means that the training set defined in this way is used for training both the network (obviously the anomalous timestamp are not used to train the first network in order to learn the correct behavior

of the sensors). Moreover, the test set contains samples that the architecture has never seen during the training part.

To enable the network to predict the behavior of the sensors given a time window, we have created a *DataLoader* object that returns n consecutive timestamps (i.e., $n = 5$) and the $n + 1^{th}$ timestamp.

Notice that no exception is considered for sliding window containing holes. As mentioned above, the dataset is divided in six different parts, moreover each part is divided into three different sets. It implies that there maybe some sliding windows where the timestamps are 18000, 18001, 26266, 26267, 26268 and we have to predict the behavior of the sensors at timestamp 26269. The number of the sliding window items containing this behavior is 25 for each set.

To train the first network, the network receives in input the 5 previous timestamps and tries to predict the next behavior of the sensors. Then the prediction is compared with the $n+1^{th}$ timestamp using the loss function.

To do it the dataframe containing the non-anomalous timestamp is used. Moreover, if a time window (of length 5) contains a number of anomalous timestamps greater than the number of the non-anomalous timestamps, then this entire time window is not used. So for example if there is an anomaly at time 1, 2 and 3, but there is not an anomaly at time 4, 5 and 6, there might be a time window that starts at time 1 and ends at time 5 and it is necessary to predict the behavior of the sensors at time 6: this last case has been discarded because the number of the anomalous timestamps is predominant and this could negatively influence the training.

Altogether we lost more or less 300 samples.

The second network is trained using the entire dataset. It is trained by receiving in input the predicted behavior of the sensors and the registered behavior at time $n + 1$. It is trained using the Cross Entropy loss function.

6. Results

The first network returns a mean squared error of 0.025. The second one has reached an F1-score of 0.997. Figure 4 shows the confusion matrix of the final results on the test set.

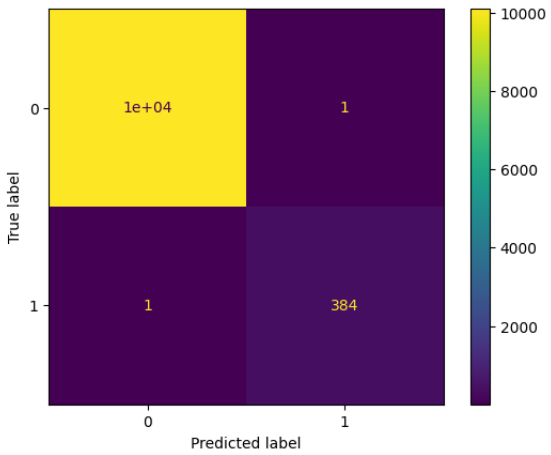

Figure 4: Confusion matrix of the final results on the test set.

7. Conclusions

In this work, we have described our artificial intelligence framework that is able to predict anomalies in time series data collected from the sensors at INFN CNAF data center. The sensor behavior remains relatively stable over time, facilitating predictability. Additionally, algorithmic assignment of labels streamlines the classification task. Obtaining feedback from experts at the INFN CNAF data center is of paramount importance.

The results of all models are extremely good. Moreover, with the weights used in the cross entropy function, models can also learn from an unbalanced dataset: 3.6% of the rows are considered anomalous.

In the future, we intend to explore the usage of graph neural network solutions for anomaly detection in time series.

References

[1] E. Ronchieri, L. Giommi, L.B. Scarponi, L. Torzi, A. Costantini, D.C. Duma et al., *Anomaly Detection in Data Center IT & Physical Infrastructure*, *EPJ Web of Conferences* **295** 07004.

[2] M. Ester, H.-P. Kriegel, J. Sander and X. Xu, *A density-based algorithm for discovering clusters in large spatial databases with noise*, in *Proc. of 2nd International Conference on Knowledge Discovery and*, pp. 226–231, 1996.

[3] T. pandas development team, *pandas-dev/pandas: Pandas*, Feb., 2020. 10.5281/zenodo.3509134.

[4] J.D. Hunter, *Matplotlib: A 2d graphics environment*, *Computing in Science & Engineering* **9** (2007) 90.

[5] M.L. Waskom, *seaborn: statistical data visualization*, *Journal of Open Source Software* **6** (2021) 3021.

[6] P.T. Inc., "Collaborative data science." `https://plot.ly`, 2015.

[7] P. Virtanen, R. Gommers, T.E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau et al., *SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python*, *Nature Methods* **17** (2020) 261.

[8] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel et al., *Scikit-learn: Machine learning in Python*, *Journal of Machine Learning Research* **12** (2011) 2825.

[9] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel et al., *API design for machine learning software: experiences from the scikit-learn project*, in *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pp. 108–122, 2013.

[10] Wikipedia contributors, *Correlation — Wikipedia, the free encyclopedia*, 2023.

[11] Pandas, "Api reference: pandas.merge_asof." https://pandas.pydata.org/docs/reference/api/pandas.merge_asof.html, 2024.

[12] scikit learn, "Api reference: Minmaxscaler." https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html, 2024.

[13] L. Xie, D. Pi, X. Zhang, J. Chen, Y. Luo and W. Yu, *Graph neural network approach for anomaly detection*, *Measurement* **180** (2021) 109546.

[14] S. Brody, U. Alon and E. Yahav, *How attentive are graph attention networks?*, 2022.

[15] S. Hochreiter and J. Schmidhuber, *Long short-term memory*, *Neural computation* **9** (1997) 1735.

[16] S. Ioffe and C. Szegedy, *Batch normalization: Accelerating deep network training by reducing internal covariate shift*, in *Proceedings of the 32nd International Conference on Machine Learning*, F. Bach and D. Blei, eds., vol. 37 of *Proceedings of Machine Learning Research*, (Lille, France), pp. 448–456, PMLR, 07–09 Jul, 2015, https://proceedings.mlr.press/v37/ioffe15.html.