DOI: 10.15439/2020F159
ISSN 2300-5963 ACSIS, Vol. 21

Network Device Workload Prediction: A Data Mining Challenge at Knowledge Pit

Andrzej Janusz*†, Mateusz Przyborowski*†, Piotr Biczyk†, Dominik Ślęzak*†
*Institute of Informatics, University of Warsaw, Warsaw, Poland
†QED Software, Warsaw, Poland

***Abstract*—We describe the 7th edition of the international data mining competition held at Knowledge Pit in association with the FedCSIS conference series. The goal was to predict workload-related characteristics of monitored network devices. We analyze solutions uploaded by the most successful participants. We investigate prediction errors which had the greatest influence on their results. We also present our own baseline solution which turned out to be the most reliable in the final evaluation.**

I. Introduction

The topic of the *FedCSIS 2020 Data Mining Challenge* falls into a category of implementing data analysis techniques in operational processes that employ either mechanical or electrical units. Part of the field, known as predictive maintenance, focuses on minimizing downtime and associated costs related to such units. Various techniques are applied to gather relevant data. It can be done using existing in-process sensors and system logs, or sensors introduced purely for predictive maintenance purposes. Data can also be acquired in an active way by injecting a test signal into a system [2].

Such data can be explored using various methods ranging from condition-based qualifiers to machine learning. A common approach relies on prediction of quantitative indicators, their association with given maintenance issues, and determination of their relationship to operational costs and failure risks [8]. The specific direction of analysis depends on individual processes under scrutiny, e.g. variability of their parameters, or importance of anomaly detection versus long-term trend detection. It often requires an ensemble of methods in order to tackle operational issues in a reliable way.

Due to their complexity, the predictive maintenance tasks are premium example of problems that could be solved using crowdsourcing, e.g. via online machine learning competitions. Rise of this approach has been accelerated thanks to popularity of sites like *Kaggle* or *Knowledge Pit*. In its core, a competition method can be drilled down to: formulation of research problem, data preparation by an unbiased team, creation of competition baseline, data analysis and solution preparation by competition participants, and finally evaluation of submissions on a platform that supports fair environment. If all these steps are provided, then the owner of data (usually the main stakeholder interested in the competition outcomes) can expect various benefits, ranging from proof of feasibility, through obtaining insights on how to resolve the problem in real-world application, up to using competition results as a guideline for assembling a dedicated R&D team.

Research co-funded by the Polish National Centre for Research and Development in frame of project MAZOWSZE/0198/19 and by QED Software.

In this paper, we investigate the outcomes of the considered challenge with a particular focus on the most substantial errors in predictions sent by participants. In Section II, we describe the challenge objectives, data sets that we prepared, the selected evaluation function, as well as our baseline model which turned out to be the most robust among all submitted solutions. In Section III, we provide an overview of the competition results and we present conclusions drawn from the analysis conducted on the set of over 700 solutions. We summarize the challenge and the paper in Section IV.

II. Competition Outline

The challenge took place at Knowledge Pit[1]. The data was provided by *EMCA Software*, the company specializing in log analytics. The goal was to predict long-term workload-related characteristics of devices, based on their history. Thus, competition results relate to EMCA's business model. Moreover, we wanted to foster deeper understanding of predictive maintenance nuances in the data science community.

A. Data preparation

The competition was held on real, previously unpublished data gathered from 3728 network devices monitored by EMCA as part of their operations. An additional, quite inspiring difficulty arose from the fact that those devices were not uniform. Logs covered readings from various types of hardware. There were also cases of different hostnames being a part of the same network, thus making their workload states correlated.

The data was collected over a period of December 2019 – February 2020. The raw data was provided in batches corresponding to individual days. Each batch contained ≈ 2.45 GB of data extracted from network device logs (≈ 220 GB in total), in form of a collection of JSON entries. Every entry consisted of device identifier, timestamp, and a list of one or more tuples indicating one of 45 so-called metrics.

Figure 1 shows the first preprocessing step. Each batch was streamed due to its large size. Individual JSON entries were parsed. The extracted hostnames were anonymized using a dynamically extended dictionary. The rest of information was transformed into EAV format (metric-timestamp-value) and aggregated for every metric in consecutive one-hour-long windows. Each such period for a given metric/hostname combination was characterized by some aggregate measures and written down to far smaller files. Similar window-based summaries are widely used in data analytics, e.g. to improve representation [7] or decrease data footprint [1].

[1]https://knowledgepit.ai/fedcsis20-challenge/

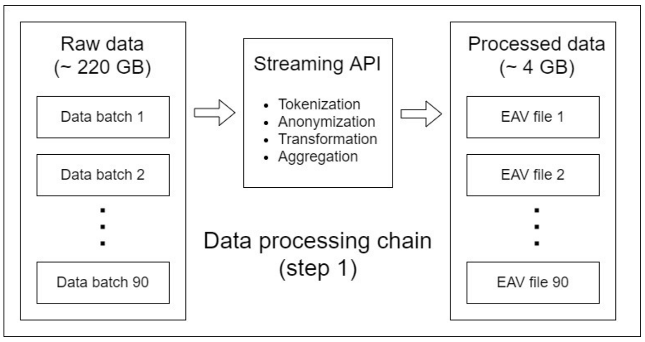

Fig. 1. The initial data preprocessing schema. The input files were processed in a streaming fashion to limit the required computational resources.

In the second step, the data from local files was merged and time series with too low number of entries were filtered out. At this point the total data size was relatively small ($\approx 4GB$). At the end, the data set was divided into training and test parts based on time. The last seven days were used as the test period. Therein, we included 10000 time series (hostname/metric combinations) which did not have any missing values in the test period. From this set, we randomly selected 1000 series to be used for preliminary evaluation of solutions.

B. Task description and evaluation procedure

The training data was made available to teams in form of a CSV file containing 10 columns. The first three of them create a joint identifier, followed by seven window-based aggregations forming in particular a *candlestick* representation of time series. The detailed column meanings are:

1) *hostname*: anonymized ID of device
2) *series*: name of the considered metric
3) *time_window*: timestamp indicating window start
4) *Mean*: the mean of the considered metric values[2]
5) *SD*: standard deviation of the considered metric
6) *Open*: the first reading in the given time window
7) *High*: the maximum reading in the given window
8) *Low*: the minimum reading in the given window
9) *Close*: the last reading in the given window
10) *Volume*: total number of corresponding readings

For each hostname in the training data, values could be arranged into series spanning for over 80 days. However, in many series, some values (time windows) were entirely missing, which typically means that a device was not accessible.

Participants were asked to predict 168 future values (i.e. hourly mean values of a given metric in one full week) of a number of devices. They were submitting their predictions to Knowledge Pit via the online evaluation system. IDs of devices and metrics were indicated in an exemplary solution file.

During the challenge (i.e. before closing it), submissions were evaluated on the already-mentioned subset of 1000 (out of 10000) test time series. We used R^2 measure, i.e. for each time series, the forecasts were compared to ground truth values, and their quality was assessed by the following formula:

[2]Values of all columns 4-10 are calculated for the considered metric (*series*) of the given device (*hostname*) within the considered *time_window*.

TABLE I. SELECTED FINAL AND PRELIMINARY RESULTS. THE SCORES OF TOP 3 TEAMS WITH REGARD TO THE PRELIMINARY AND FINAL SCORES ARE SHOWN. THE LAST COLUMN INDICATES THE NUMBER OF SUBMITTED SOLUTIONS. NOTICEABLE IS THE NEGATIVE CORRELATION BETWEEN THE NUMBER OF SUBMISSIONS AND FINAL SCORE OF PARTICULAR TEAMS.

Rank	Team name	Preliminary	Final score	#subs
1	baseline solution	0.2267	0.229530	3
2	Les Trois Mousquetaires	0.1888	0.162979	19
3	papiez69	0.1841	0.151499	13
4	Wrong Team Name	0.1836	0.143708	6
...	...	...	...	...
13	cdata	0.2766	-0.059837	90
14	amy	0.3130	-0.138349	100
...	...	...	...	...
17	Dymitr	0.3223	-0.779576	146
...	...	...	...	...

$$R^2(f,y) = 1 - \frac{RSS(f,y)}{TSS(y)} \tag{1}$$

where f is a vector of forecasts, y is the target ground truth, $RSS(f,y) = \sum_i (y_i - f_i)^2$ is the residual sum of forecast squares, and $TSS(y) = \sum_i (y_i - \hat{y})^2$ is the total sum of squares, where $\hat{y}$ is the mean value of time series y estimated using the available training data. The submission score is the average R^2 over all time series from the test set.

The best evaluation results of participating teams were visible on the public Leaderboard. Each team could submit up to 100 solutions, but teams could merge during the competition. Thus in the end, the total number of submissions for a given team could be greater than this limit (in such a case, the merged team could not submit any new solution files).

The final evaluation was performed after completion of the competition using the remaining part of the test data (90%). Those results were published online too. Only those teams who submitted a report describing their approach before the end of the challenge were qualified for final evaluation.

C. Our baseline solution

At the beginning of the challenge, we prepared a relatively simple baseline solution in order to provide to participants a reference score at the Leaderboard. We did also for the purpose of equipping EMCA with a light-weight tool for detecting anomalies in device usage patterns in real-time.

First, we cleaned the training data out of outlying metric readings. Such readings could be a result of some unexpected device malfunction, monitoring software error, or some unusual actions performed by device users. Since we were mostly interested in detecting typical device usage patterns, outliers in the training data could distort our forecasts.

There are many approaches to time series anomaly detection. In our solution, we used the well-known *3-sigma* method. We assumed weekly periodicity of the considered time series, which was confirmed on a random sample using the Fisher's test. We divided the training data into disjoint, one-week long time windows. For each window, we estimated the mean and standard deviation. We clipped the time series values which were identified as outliers by means of 3-sigma. We treated the clipped windows as time series motifs [5].

Then, for each time series in the test data, we extracted the latest weekly time window from the training data as a tempo-

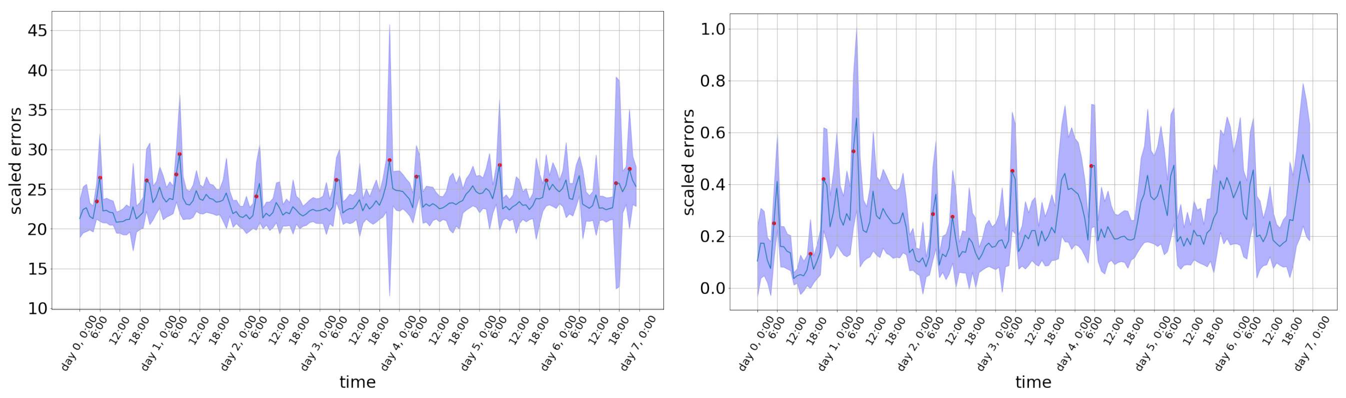

Fig. 2. Squared differences between forecasts and reference values. On the left, scaled by TSS of each series, aggregated by the metric/hostname combination, and averaged for each hour. The shaded area represents a distance of one standard deviation from the mean. Red dots indicate an increase in the error rate above one standard deviation from the mean of all errors. The plot on the right shows the resulting values with additional scaling of each series to [0,1] interval.

rary validation set. We averaged the last three motifs from the remaining time windows to create the series templates. It means that – denoting the last i-th motif for the k-th series by $\vec{x}_{k,i}$ – the corresponding template is specified as:

$$T_k = \frac{1}{3}\sum_{i=1}^{3}\vec{x}_{k,i} \tag{2}$$

We compared such templates with the corresponding validation windows by means of R^2 score. Then we updated templates with the data from validation period. When making forecasts, we applied templates with $R^2 > 0$ to predict the corresponding series in the test period. Otherwise, our forecast was simply the global mean for the given series, estimated on the whole of training data. Table I indicates that this method achieves the best final score among all submitted solutions.

III. Competition Results

The competition attracted 151 teams from over 30 countries. The highest number of participants had IP addresses from Poland (52), India (19), Russia (15), China (7), and the United States (7). Over 700 correctly formatted solutions were submitted. Besides the above-discussed baseline, Table I reports the final ranks, scores, and the number of submissions for some of the best performing teams. More details about some of those teams can be found in [3], [6], [9], [10].

The best submitted solution, i.e. Rank 2 in Table I, relied on an ensemble of XGBoost, Prophet and linear regression models. The final model took into account their individual performance for each host, applying the mean value if neither of them proved to give satisfactory results, or if there were less than 300 of data points for the host. Rank 3 utilized ensemble of two different XGBoost models – one using only hours and holiday days as features, and the other using also days of the week – together with the mean value for the cases for which the considered models delivered R^2 less than 0.04.

A. Error distribution over time

As device logs vary greatly, we run our comparisons using scaled and averaged values. Figure 2 suggests that reasoning about further-future values does not deteriorate over time. This may indicate that submitted methods can successfully (on average) deal with data seasonality, grasping periodicity within the series that may be observed in the training data set and, based on that, producing future forecasts. On the other hand, the errors tend to happen around the same hours every day. Such events may be unpredictable, although the highest final score solutions are more robust with this respect.

B. Error distribution among the best solutions

At Knowledge Pit, competition participants can verify their solutions using a small test data portion. This functionality usually helped in fine-tuning meta-parameters of developed methods. However this time, the solution with the highest preliminary score did not want to generalize to the entire test set at all. According to Figure 3, the preliminary-best solutions tend to neglect the importance of some series which turn out to be a significant part of the test data. This may be because of under-representation of some patterns and overall difficulty in forecasting those series in the training data set.

C. Errors according to the metrics

Figure 4 shows that metrics vary in terms of forecasting difficulty. This fact could be a reason for some of participating teams to give up some series and focus on increasing R^2 based on a few simpler ones. Accordingly, let us note that our way of splitting test data onto preliminary and final subsets preserved most of desired statistical properties of the metrics aggregations. Out of 27 distinct metrics in the test set, 25 occur in preliminary part, with sufficient cardinality. This yields a corollary that it was not a design of data sets that caused problems, but the very structure of explored data.

IV. Conclusions

We reported our international challenge aimed at predicting long-term network device workload characteristics. Our baseline model achieved the score $R^2 = 0.23$ and we assess that this result can be further improved. The analysis of errors of other submitted solutions revealed no significant correlation with a time horizon of forecasts. It suggests that models can often correctly take into account periodicity, but any deviations are hard to predict. It not only means that modeling of typical workload patterns is feasible, but it may also allow us to design a simple real-time anomaly detection algorithm.

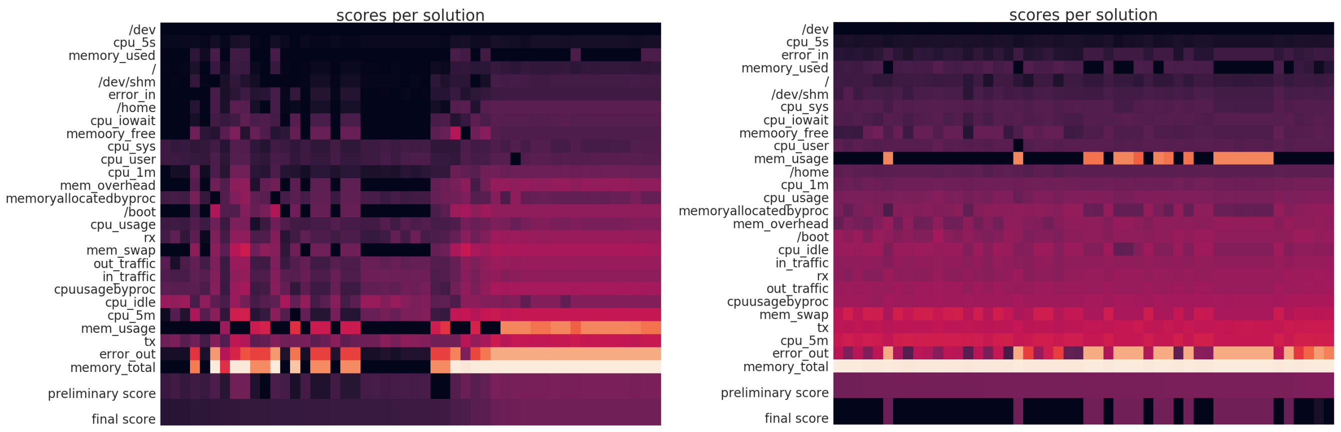

Fig. 3. R^2 scores per submission (the darker the color, the lower the value of R^2 for particular submitted solution on data corresponding to a particular metric), for individual metrics (*series*). Vertically, rows are sorted by the mean score (increasing towards bottom). Horizontally, solutions are sorted by the final score. The plots on the left and right sides show the values for the top 75 submissions with regard to the final and preliminary scores, respectively.

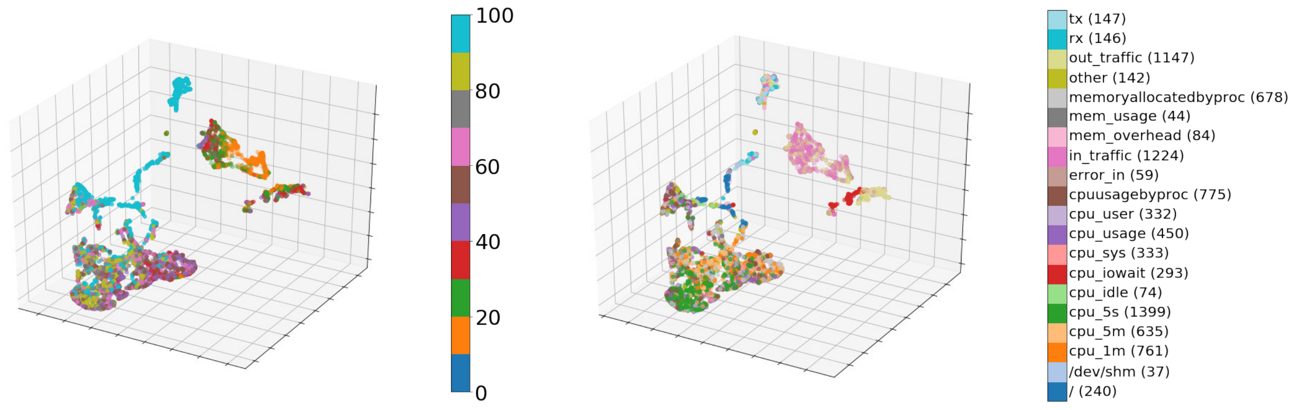

Fig. 4. Errors (9000 series, each 704 dimensions) reduced with UMAP [4] to three dimensions. The plots color the points according to the metric (on the right) and to percentiles of the mean values (on the left). The points form clusters by means of a similar scale of values and metrics.

It is worth discussing why competition participants – representing truly diverse background and data science experience levels – could not come up with any solution yielding better prognostic power than our baseline. One reason is that we focused on typical work patterns of the system, not attempting to forecast outliers. From the perspective of R^2 evaluation score, any model that did try to predict anomalies was highly punished for missed forecasts. It may mean that although – as mentioned above – it seems to be easy to detect anomalies, it is incomparably harder to predict their values.

Future works can include better adjustment of the applied error function to end-user expectations. Moreover, one can develop an ensemble approach combining models tuned to typical workloads (e.g. our baseline) with those reflecting outliers (some other submissions). One can also consider anomalies of various kinds: power-law anomalies (*black swans*), phase transition anomalies (*dragon knights*), and unique events not exceeding operational parameters (*unicorns*).

Finally, as for our general competition-based approach to crowdsourcing of data mining problems, one can see that the current setup at Knowledge Pit fits both, the needs of data science community and expectations of real-life data owners. Accordingly, new challenges will be organized.

References

[1] A. Chądzyńska-Krasowska and M. Kowalski. Quality of Histograms as Indicator of Approximate Query Quality. In *Proc. of FedCSIS 2016*, pages 9–15.

[2] H. M. Hashemian. State-of-the-Art Predictive Maintenance Techniques. *IEEE Trans. Instrum. Meas.*, 60(1):226–236, 2011.

[3] C. Liu. Shallow, Deep, Ensemble Models for Network Device Workload Forecasting. In *Proc. of FedCSIS 2020*.

[4] L. McInnes, J. Healy, N. Saul, and L. Großberger. UMAP: Uniform Manifold Approximation and Projection. *J. Open Source Softw.*, 3(29):861, 2018.

[5] A. Mueen and E. Keogh. Online Discovery and Maintenance of Time Series Motifs. In *Proc. of KDD 2010*, pages 1089–1098.

[6] D. Ruta, L. Cen, and Q. H. Vu. Deep Bi-Directional LSTM Networks for Device Workload Forecasting. In *Proc. of FedCSIS 2020*.

[7] Ł. Sosnowski and T. Penza. Generating Fuzzy Linguistic Summaries for Menstrual Cycles. In *Proc. of FedCSIS 2020*.

[8] G. A. Susto, A. Schirru, S. Pampuri, S. McLoone, and A. Beghi. Machine Learning for Predictive Maintenance: A Multiple Classifier Approach. *IEEE Trans. Ind. Informatics*, 11(3):812–820, 2015.

[9] T. Wittkopp, A. Acker, S. Nedelkoski, J. Bogatinovski, and O. Kao. Superiority of Simplicity: A Lightweight Model for Network Device Workload Prediction. In *Proc. of FedCSIS 2020*.

[10] M. Zuefle and S. Kounev. A Framework for Time Series Preprocessing and History-based Forecasting Method Recommendation. In *Proc. of FedCSIS 2020*.