POLITECNICO MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE E DELL'INFORMAZIONE

Executive Summary of the Thesis

Machine-Learning-Assisted Failure Prediction in Microwave Networks based on Equipment Alarms

Master Degree in Telecommunication Engineering - Ingegneria delle Telecomunicazioni

Author: Francesco Lateano

Advisor: Prof. Francesco Musumeci

Co-advisors: Prof. Massimo Tornatore, Dr. Omran Ayoub

Academic year: 2020-2021

1. Introduction

Failure management plays a role of capital importance in microwave networks to avoid service disruptions and to satisfy customers' service level agreements. Currently, failure management in microwave networks strongly relies on the ability of domain experts who perform failure troubleshooting observing devices alarms and performance metrics retrieved from the network. Machine learning (ML) [1] [2] [3] promises to revolutionize this approach, by introducing automated methods for failure management [4]. In this work, we describe a successful framework of Machine Learning (ML) for automatic failure prediction in microwave networks based on real-field equipment alarms.

Modern microwave equipment are now installed with built-in monitoring capabilities, and they are capable to generate a large amount of data, which can be leveraged to automate microwave failure management using ML [5]. Indeed, this large amount of data collected through such monitors can now be stored and elaborated in centralized locations thanks to new advanced control management solutions, as network telemetry, SDN and/or orchestration frameworks that, thanks to modern network intelligence (computing capabilities) can be placed virtually everywhere (e.g., leveraging Network Function Virtualization and/or Mobile Edge Computing). Despite that troubleshooting on alarms logs is mainly hand-made by domain experts, resulting in a limited classification of available data. Hence, as first step a data augmentation implementation via unsupervised learning is applied to unlabelled samples. Second, automated failure-cause prediction in microwave networks, allowing operators to reduce service unavailability, is performed. Third, the most likely failure root-cause is predicted, so appropriate countermeasures can be effectively put in place (e.g., by choosing an in-field intervention vs. a remote equipment reconfiguration), potentially ahead of the actual fault.

Recalling that on microwave links, different heterogeneous causes (e.g., adverse atmospheric conditions, or obstacles) lead to service unavailability and produce alarms activation and that failure identification is traditionally accomplished by domain experts via direct inspection of alarms logs (most of the time after failure events), our main contribution is to provide a prediction methodology to anticipate and speed up identification of fault causes leveraging on automation via ML technology, resulting in a trade-off between two approaches with different temporal scales, one more accurate short-term but with computational constraint limiting prediction horizon and one long-term able to achieve longer predictions but losing accuracy and the ability of short-term predictions.

1.1. Problem Definition

In this work, we consider source data from network equipment providing information on alarms behavior, but not on specific failure-causes, which can be even unknown for network operators, so high-level problem of predicting future root-causes faults is broken down into the following functional objectives:

1. The first objective is to group data in order to extract significant information from alarms data and identify distinct failure-causes associated to alarms behavior;
2. Moreover, forecast alarms statuses is our second objective;
3. Finally, detection of failures and identification of the relative cause, are our final goals.

2. Machine-Learning Failure Prediction

This chapter details the methodological approaches and algorithms used to perform fault prediction in microwave networks. We start with data augmentation via kMeans in order to increase ground truth for model training. Then, we describe two approaches: first we propose a short-term multi-step prediction framework with 1 second granularity, compost
ed by a predictive deep learning model applied to bitsequence time-series to forecast alarm statuses in the future [6], [7]. Then, failure detection system and fault detection module using machine-learning ensembles is described. Beside we propose a long-term single-step approach, with 15 minutes granularity, for direct fault prediction via machine learning techniques. The overall failure management framework developed in this work is represented by the diagram 1, describing the logic workflow of our work in a holistic way and summarized by the following blocks:

1. Data Retrieval, it includes all operations related to data gathering, analysis and preparation, including data augmentation with kMeans;
2. Training and Validation, it considers machine learning operations as model definition, training and validation, including input data pre-processing with regards to ML models requirements. Models used are LSTM, kNN, SVM, RF, NN.
3. Prediction is performed following two alternative approaches, i.e., short-term multi-step or long-term single-step; the first consists of alarms forecast, detection and identification; the latter consists of sole root-cause forecast.

2.1. Dataset description

Datasets provided by SIAE Microelettronica used for our analysis is a collection of logs from AGS20 equipment type, a universal microwave aggregation platform. The input data consists of collected measurements and alarm logs which are used to analyze network failures, obtained by an Italian microwave network of 10841 radio links. In this study, we concentrate on alarm logs to determine hardware failures. We analyze network alarms for one week period starting from the beginning of 27/09/2019 to the end of 4/10/2019. We generate various datasets with different alarm structure. First, bitsequence dataset consists of a log with information on alarms ON/OFF status, also including reference to equipment and site, and additional information related to the alarm. Second, alarm statistics windows that include all links and all the existing alarms in the system, computing occurrences with a 15 minutes granularity.

Meanwhile, time slices to feed the alarms forecasting model are created, where a slice is a sequential bitsequence subset with length equal to the sum of 2 parameters: forecaster input size and forecaster output size [8]. So, dataset is transformed into a new dataset constituted by a set of *slices* that feeds the forecast model.

Finally, our implementation produces a dataset where each window is labelled with a root-cause label.

2.2. Short-term Multi-Step Approach

In this approach, prediction of future fault root-causes is divided into 3 sequential modules: alarm forecasting, failure detection and fault identification. Its workflow is shown in Figure 2 and is summarized with the following steps:

1. Alarms are collected and organized in bitsequence per each link;
2. Bitsequence are prepared for forecasting: data is splitted into training, validation and testing set. Then, each dataset is sliced into multivariate, i.e. considering multiple alarms, time-series with simulation scenario dimensions - as defined in next paragraphs;
3. Alarms forecast is performed;
4. Windows are prepared from predicted alarms bitsequence;
5. Failure detection and identification is performed on windows using machine learning classifier.

2.3. Long-term Single-Step Approach

We propose another approach to perform failure root-cause identification leveraging on data structure characterization, as described in the following steps:

1. Applying data augmentation to collected windows including no-failure case creating as output a windows dataset with either relative failure root-causes or no-failure.
2. Then, fault label is updated of the prediction time horizon . e.g. 1 hour prediction consists of updating the current label with the label of the next fourth window, since 1 hour is 4 windows.
3. Finally, this *swapped* dataset is used to train previously defined model for fault identification, so the resulting model could be used for future prediction of faults.

Figure 3 shows its workflow diagram.

2.4. Hyperparameters Optimization

This section provides first a definition of used algorithms parameters used in execution of our framework, as summarized in Tables 1, 4, 3, 2 and 5. At the end, a description of the performance computation method is provided.
Tables 1, 2, 3, 4, 5 summarize models parameters chosen.

Parameter	**Value**
Output Layer	Dense with sigmoid
Loss function	Binary cross entropy
Optimizer	Adam
Metrics	TP, TN, FP, FN, Acc, Rec, Prec, AUC, PRC
Overfitting	Earlystopping with patiente = 10
Epoch	1 (trained iteratively on links)

Table 1: Hyperparameters for LSTM

Parameter	**Value**
Number of trees	10
Maximum tree depth	10
Minimum number of split	2

Table 2: Hyperparameters for RF

Figure 1: Overall Prediction Diagram

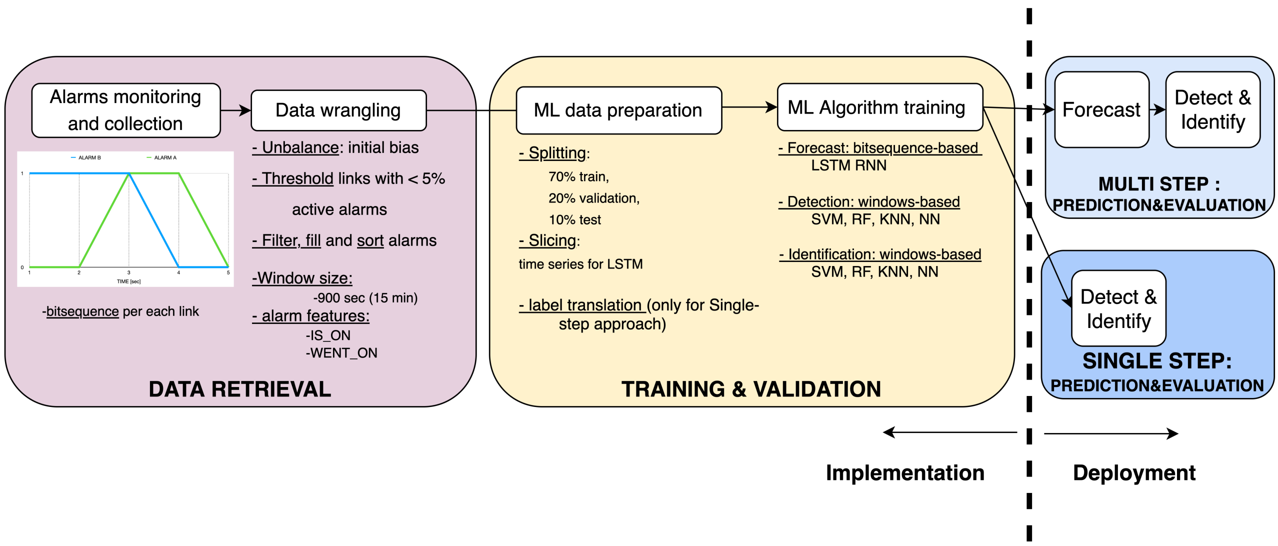


Figure 2: Short-Term Multi-Step Approach

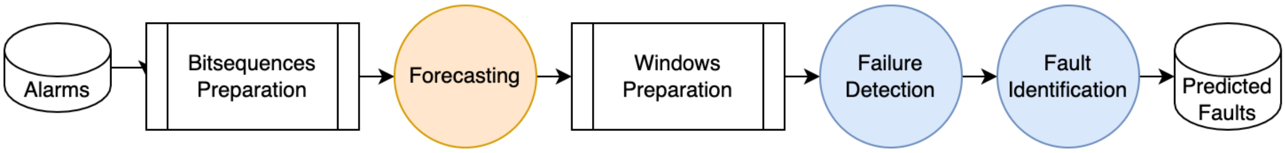
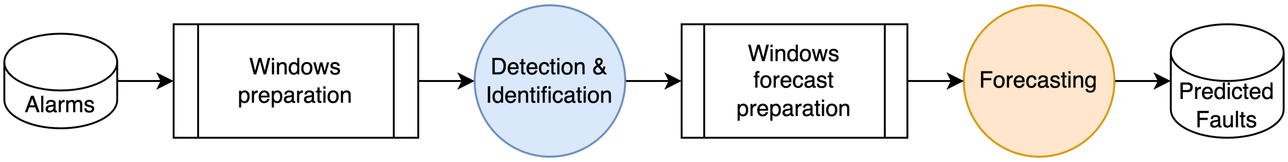


Figure 3: Long-Term Single-Step Approach



Parameter	Value
Number of neighbors	5
Weight function	Distance
Distance function	Manhattan
Searching algorithm	Kd-tree

Table 3: Hyperparameters for KNN

Parameter	Value
Regularization parameter	100
Kernel	RBF
Kernel coefficient	0.01
Decision function	One-versus-rest

Table 4: Hyperparameters for SVM

Parameter	Value
Number of hidden neuron	5
Activation function	Linear

Table 5: Hyperpar. selected for ANN algorithm

Now, the classical method for estimating the accuracy of a ML algorithm is k-folds cross-validation (CV).
The simplest measure of the quality of a classifiers is the accuracy, i.e., the proportion of instances which are correctly classified. However, accuracy assumes that all errors are equally bad, while usually different types of error have different costs. So recall and precision are more relevant metrics than accuracy: recall = number of positive cases predicted as positive total number of positive cases precision = number of positive cases predicted as positive number of positive predictions Precision and recall are two contrasting objectives and different algorithms give different trade-offs on these measures.
In our study, we divided sequentially bitsequence data to 70% as training set and 20% as validation and 10% as test. Framework performance is measured in terms of accuracy, precision, recall. In the following paragraphs results are presented for each phase.

3. Results

Our experiments were conducted on a PC with Processor: Intel Core i7 processor and 16 GB RAM. Results are the following:

3.1. Short-Term Multi-step Prediction

We simulated two main scenario for this approach: first forecasting 10 seconds in the future using 10 sec as input; second, forecasting 120 seconds using 10 seconds as input. Note that our time horizon maximum depth - 120 seconds- was limited by simulation hardware constraint, i.e. random access memory.
Comparing different time length prediction is not significant for models comparison since model performs extremely well for considered scenario. In fact, we can highlight a decrease of performance moving from 2 seconds prediction to 10 seconds, while increasing the time horizon till the limit case of 120s does not show any variation on forecast performance, as shown in Fig. 4.
Then we compare the performance of our LSTM-based predictor against two baseline forecast-

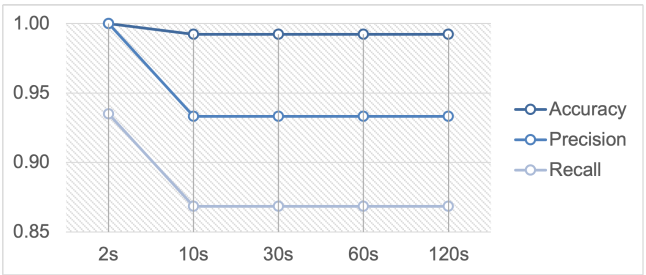


Figure 4: LSTM performance varying prediction horizon

ers. First using a naive model where each future second of alarms is predicted based on a simple probability value; and a second one with a more evoluted probabilistic model taking a linear combination of the overall bitsequence unbalance value and the positive ratio of moving window of the last seconds. For sake of brevity baselines details are omitted in this document. The accuracy, precision and recall are compared to LSTM performance in Fig 5 showing a great increase in performance value of LSTM against baselines. Moreover, the distribution of accuracy, precision and recall obtained across the different microwave links are shown in the boxplots of Fig. 6, showing the distributions of these metrics resulting in a better adherence of LSTM to each link.

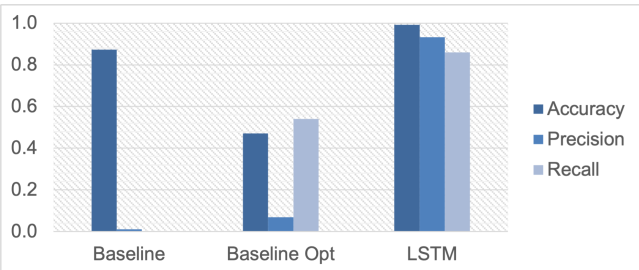


Figure 5: Alarms Forecast Performance Comparison

Complexity is linear with regard to bitsequence slice dimensions. Execution times are presented in figure 7.
The accuracy, precision and recall for detection classifiers for both scenario are plotted in Fig. 8. SVM, RF, KNN and ANN have extremely high metrics over 95%. Farther, all classifiers provide results close to each other.
Execution times considering both training and prediction are plotted on Fig. 9. Here kNN shows the lowest execution time, as expected, considering that no training is performed by the algorithm, followed by RF, while at relevant distance we have SVM and at last NN.

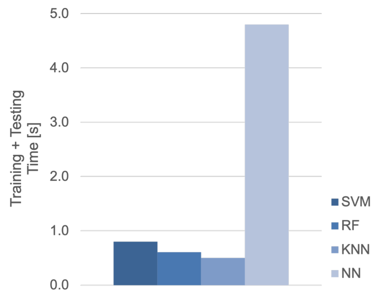


Figure 9: Detection Execution Time

Next identification module is considered with accuracy, precision and recall for classifiers and both scenario are plotted in Fig. 10. Here too SVM, RF, KNN and ANN have extremely high metrics over 95%. Farther, all classifiers provide results close to each other.
Execution times considering both training and prediction are plotted on Fig. 11. Here kNN shows the lowest execution time, as expected, considering that no training is performed by the algorithm, followed by RF, third is at SVM and at last with remarkable delay is NN.

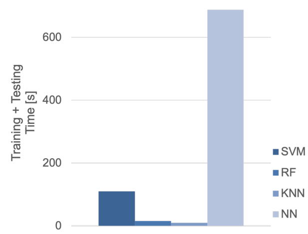


Figure 11: Identification Execution Time

3.2. Long-Term Single-step Prediction

Three prediction scenario are compared with respect to prediction interval: 1 hour, 24 hours and 48 hours.
Accuracy, precision and recall are respectively plotted in figures 12, 13, 14 pointing out a general trend inversely proportional to prediction interval, while a similar pattern appears for all metrics in different scenario. Furthermore, we have similar results for all models except in the 48 hours scenario where kNN performance plumbs of 0.2 points respect to other classifiers.

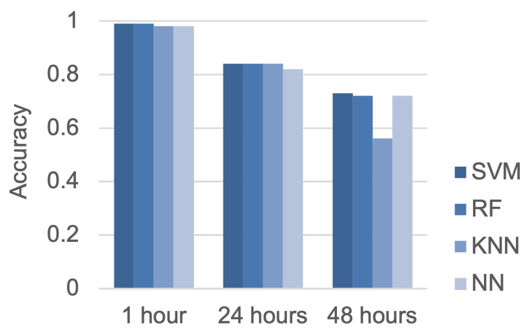


Figure 12: Single-step performance: Accuracy

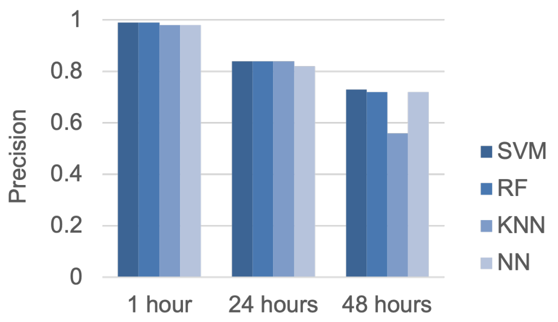


Figure 13: Single-step performance: Precision

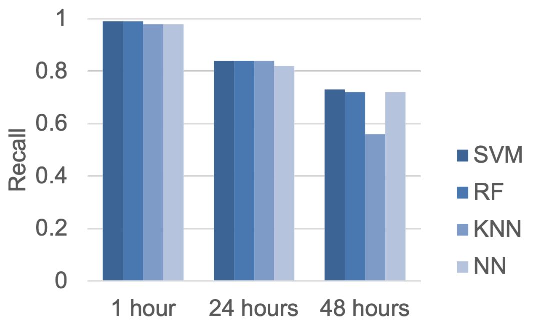


Figure 14: Single-step performance: Recall

Execution times considering both training and prediction are plotted on Fig. 15. Also here Here KNN shows the lowest execution time, as expected, considering that no training is performed by the algorithm, RF is second, at considerable distance SVM and after another time gap, NN finishes.

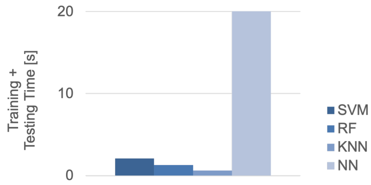


Figure 15: Single-step Time

4. Conclusions

Overall, framework results in very high model accuracy and reasonable linear time complexity, suggesting the approaches are suitable to solve this problem, despite strong dataset unbalance. Short-term multi-step and long-term single-step methods allow a exhaustive analysis and prediction on input data and, as future work, with better hardware, short term multi-step prediction horizon could be extended and compared directly with long-term single-step approach.
In addition, future works could be the integration of a graph neural network for fault identification and localization. Furthermore alternative forecast models could be benchmarked.

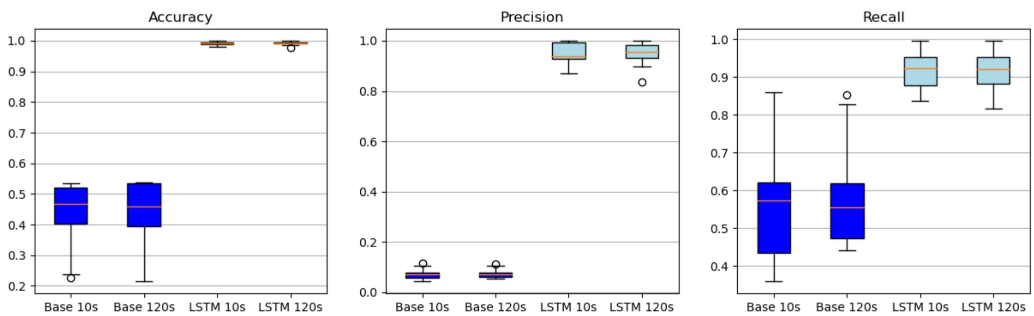


Figure 6: LSTM Performance against Baseline Opt

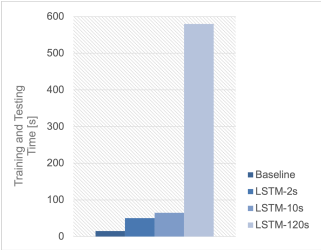


Figure 7: Forecast execution time

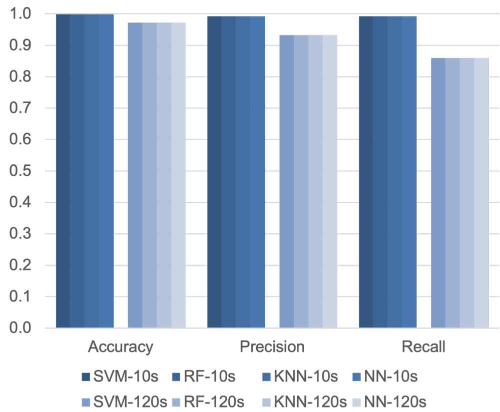


Figure 8: Detection performance

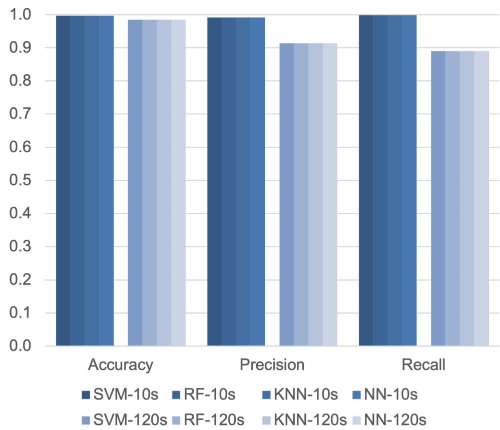


Figure 10: Identification performance

References

[1] F. Musumeci, C. Rottondi, G. Corani, S. Shahkarami, F. Cugini, and M. Tornatore, "A tutorial on machine learning for failure management in optical networks," *Journal of Lightwave Technology*, vol. 37, no. 16, pp. 4125–4139, 2019.

[2] M. Zhang and D. Wang, "Machine learning based alarm analysis and failure forecast in optical networks," in *2019 24th IEEE Opto-Electronics and Communications Conference (OECC) and 2019 International Conference on Photonics in Switching and Computing (PSC)*, pp. 1–3, 2019.

[3] G. Savva, T. Panayiotou, I. Tomkos, and G. Ellinas, "Deep graph learning for qot estimation of unseen optical sub-network states: Capturing the crosstalk impact on the in-service lightpaths," 2021.

[4] S. Barzegar, M. Ruiz, A. Sgambelluri, F. Cugini, A. Napoli, and L. Velasco, "Soft failure detection, localization, identification, and severity prediction by estimating qot model input parameters," *IEEE eTransactions on network and service management*, p. 1, 2021.

[5] F. Musumeci, L. Magni, O. Ayoub, R. Rubino, M. Capacchione, G. Rigamonti, M. Milano, C. Passera, and M. Tornatore, "Supervised and semi-supervised learning for failure identification in microwave networks," *IEEE Transactions on Network and Service Management*, 2020.

[6] X. Wang, Y. Zhang, and M. Zhang, "Time series prediction method based on variant lstm recurrent neural network," *Neural Processing Letters*, vol. 52, no. 2, pp. 1485–1500, 2020.

[7] S. Hochreiter and J. Schmidhuber, "Lstm can solve hard long time lag problems," *Advances in neural information processing systems*, pp. 473–479, 1997.

[8] Tensorflow, "Time series forecasting." `https://www.tensorflow.org/tutorials/structured_data/time_series`.