ASAP Hardware Failure-Cause Identification in Microwave Networks using Venn-Abers Predictors

Nicola Di Cicco, *Graduate Student Member, IEEE*, Memedhe Ibrahimi, *Member, IEEE*
Omran Ayoub, *Member, IEEE*, Federica Bruschetta, Michele Milano,
Claudio Passera, and Francesco Musumeci, *Senior Member, IEEE*

***Abstract*—We investigate classifying hardware failures in microwave networks via Machine Learning (ML). Although ML-based approaches excel in this task, they usually provide only hard failure predictions without guarantees on their reliability, i.e., on the probability of correct classification. Generally, accumulating data for longer time horizons increases the model's predictive accuracy. Therefore, in real-world applications, a trade-off arises between two contrasting objectives: i) ensuring high reliability for each classified observation, and ii) collecting the minimal amount of data to provide a reliable prediction. To address this problem, we formulate hardware failure-cause identification as an *As-Soon-As-Possible (ASAP) selective classification problem* where data streams are sequentially provided to an ML classifier, which outputs a prediction as soon as the probability of correct classification exceeds a user-specified threshold. To this end, we leverage Inductive and Cross Venn-Abers Predictors to transform heuristic probability estimates from any ML model into rigorous predictive probabilities. Numerical results on a real-world dataset show that our ASAP framework reduces the time-to-predict by ∼8x compared to the state-of-the-art, while ensuring a selective classification accuracy greater than 95%. The dataset utilized in this study is publicly available, aiming to facilitate future investigations in failure management for microwave networks.**

***Index Terms*—Microwave networks, failure-cause identification, As-Soon-As-Possible classification, Venn-Abers predictors**

I. Introduction

MICROWAVE networks are widely deployed as an alternative technological solution to optical backbones, especially to support backhauling of mobile traffic. Next-generation (6G) communication services supported by such networks are characterized by extreme availability requirements such as six 9s or even higher reliability [1]–[3]. Therefore, prompt failure management represents a key factor for the success of microwave networks in the 6G ecosystem. A quick and reliable hardware failure-cause identification, as well as precise discrimination of faulty devices, are of paramount importance, as the countermeasures adopted by network operators to address network failures (e.g., whether to reconfigure, repair, or even substitute a network device) strongly depend on these two factors. In this context, early hardware failure detection and cause identification not only helps in lowering mean-time-to-repair (MTTR) but also allows effective service maintenance at a higher level, e.g., achieved by rerouting traffic from a malfunctioning link towards an operating one while link troubleshooting and repairing operations take place.

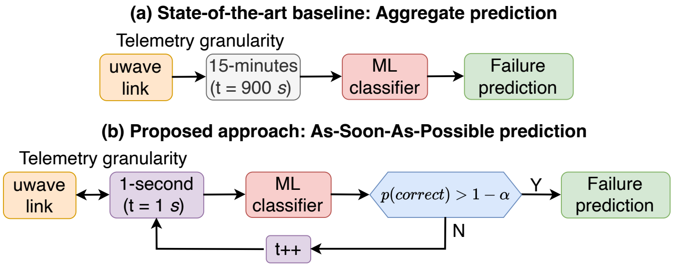


Fig. 1. (a) Aggregate prediction vs. (b) As-Soon-As-Possible prediction. Aggregate prediction collects network telemetry for a fixed-size observation window, and then feeds it to an ML-based failure classifier. In contrast, As-Soon-As-Possible prediction returns a failure prediction as soon as the probability of correct classification exceeds a user-specified threshold $1 - \alpha$.

Current hardware failure classification approaches are based on observing network parameters (e.g., transmitted/received power measures, configured modulation format, and other links settings) and equipment alarms retrieved by a Network Management System (NMS). To output a failure diagnosis and hence devise an appropriate mitigation strategy, operators rely on knowledgeable domain experts, who typically observe and analyze the retrieved information case by case. Due to the increased complexity of modern microwave networks and the large volumes of data that need to be analyzed by domain experts to extract valuable information on failure causes, there is an urgent need to automate the failure management process. To this end, Artificial Intelligence (AI) and Machine Learning (ML) are key enablers, as shown by the large amounts of studies that recently appeared in literature [4]–[8].

However, although AI/ML models achieve satisfactory predictive performance in hardware failure-cause classification, obtaining a reliable model that provides an uncertainty measure of its outputs is crucial for real deployments, as it allows operators to make informed decisions on how to handle failures properly. To reach this objective, we leverage Uncertainty Quantification (UQ) in ML models, specifically Inductive and Cross Venn-Abers predictors [9], a family of probabilistic predictors with formal validity guarantees closely related to the field of Conformal Prediction [10]. We leverage Venn-Abers predictors to develop ML-based hardware failure classifiers that output not only the most likely root cause of a given hardware failure but also the probability that

N. Di Cicco and M. Ibrahimi are co-first authors of this paper. N. Di Cicco, M. Ibrahimi, and F. Musumeci are with the Department of Electronics, Information, and Bioengineering (DEIB), Politecnico Di Milano, Italy. E-mail: {name}.{surname}@polimi.it. O. Ayoub is with the Department of Innovative Technologies (DTI), University of Applied Sciences of Southern Switzerland, Viganello, Switzerland. E-mail: omran.ayoub@supsi.ch. F. Bruschetta, M. Milano and C. Passera are with SIAE Microelettronica S.p.A., Italy.

the prediction is correct. This additional information on the predictive uncertainty can be exploited by domain experts to properly gauge the risk of making decisions based on the ML model's outputs.

Fig. 1 shows a state-of-the-art solution based on *Aggregate prediction* and our proposed solution based on *As-Soon-As-Possible prediction*, aiming to make a prediction as-soon-as-possible, once the required statistical guarantees are met. Prior works that perform Aggregate prediction [11], [12] collect fixed-size 15-minute windows of equipment alarms and feed the data to an ML classifier for hardware failure-cause prediction. Conversely, As-Soon-As-Possible prediction considers *one-second* telemetry granularity of equipment alarms, and feeds the data to an ML classifier that makes a failure-cause classification only if a statistical guarantee is met. In case not, new data from the next observation second is queried. The process is repeated until the statistical guarantee is met.

Intuitively, a reliable prediction, i.e., failure-cause classification with low uncertainty, can be obtained after a sufficient amount of information (e.g., a data stream including information on the status of multiple equipment alarms) has been collected and fed to the AI/ML model. However, this is in contrast with the ideal goal of *As-Soon-As-Possible* (ASAP) prediction, where an operator wishes to react to failures as soon as they occur or, more realistically, with a minimal amount of information (and hence, after a minimal amount of time) sufficient to obtain a reliable prediction. As a motivating example, we quantify the performance gap between classifying a hardware failure at the first second and classifying at the end of a 15-minute window on our alarms dataset (described in detail in Section III). Considering a state-of-the-art XGBoost [13] model, failure classification at the first second and at the end of the 15-minute window result in cross-validated accuracy of $88\% \pm 2\%$ and $96\% \pm 2\%$, respectively. We conclude that, though the first-second classification already yields a very good performance, the gap with the 15-minute classification is practically significant. In particular, in the context of failure management, it is paramount to achieve near-perfect failure classification, as predicting the wrong failure cause may result in choosing inappropriate or dangerous mitigation strategies.

Therefore, the research question we aim to address in this paper is: *Can we autonomously return a maximally accurate hardware failure-cause prediction in the least amount of time?*

To solve the above problem, we propose an ML-based hardware failure-cause classification framework that returns a prediction as soon as the probability of correct classification is greater or equal to a user-specified threshold (e.g., 95% or 99%). In other words, our proposed framework can provide failure-cause predictions that are both *timely* and *highly reliable*. Compared to our prior work [14], the main novelties proposed in this paper can be summarized as follows: 1) we explicitly address a reliability aspect of probabilistic prediction, while our prior work only considered hard predictions, and 2) we redefine the problem by considering ASAP failure-cause classification at one-second granularity, while in the prior work, we considered fixed-size observation windows to forecast alarm states and corresponding failure causes without any statistical guarantee. Our key contributions are summarized as follows:

- We introduce a new dataset for ML-based failure management, comprising alarms and ground-truth annotations indicating failure causes in microwave links from a real-world microwave network, and make it publicly available to the research community.[1] (Section III)
- We propose a principled methodology for ML-based *As-Soon-As-Possible* hardware failure-cause classification in microwave networks, such that the ML model retrieves from the network the minimal amount of information on device alarms to output a prediction only when its probability of correct classification is at least above a user-specified safety threshold. To achieve this, we leverage Venn-Abers predictors, which offer theoretical guarantees on predictive probabilities. (Section IV)
- We validate our approach against the current state-of-the-art, illustrating that our approach consistently yields better probabilistic predictions, thereby allowing for reliable selective classification. Furthermore, we explore and discuss tunable performance trade-offs introduced by our proposed methodology. (Section V)

II. Related work

The application of ML for failure detection and failure-cause identification in telecommunication networks is receiving considerable attention, as it offers operators the ability to take mitigation actions promptly [15]–[17]. In this section, we discuss some recent literature utilizing ML for failure management with a specific focus on microwave networks.

Prior works utilized ML for detecting failures due to transmission parameter degradation or attenuation on microwave links [18]–[20]. For instance, in [18], authors propose an ML-based approach for continuously monitoring the performance of a microwave link and detecting degradation due to natural weather conditions, leveraging on performance measurements, such as signal strength and signal-to-noise ratio, and weather information. Another work [19] proposes an approach based on Long Short-Term Memory (LSTM) and recurrent neural networks to continuously predict rain-induced attenuation due to weather conditions using past measurements. Similarly, authors in [20] propose various ML-based approaches for real-time analysis of the link's performance and forecasting of rain-induced attenuation leveraging historical data.

Other works have utilized data available from microwave links to predict a broader set of failures. For instance, [21] proposes supervised and semi-supervised learning approaches for failure-cause identification leveraging link performance data from a nationwide microwave network. Gathered data measurements were aggregated in 15-minute intervals, and ML techniques were employed to train models to identify six categories of failure causes, achieving classification accuracy up to 95%. In [22], authors focus on tackling the same problem considering, in addition to the link's performance measurements, alarm data stemming from devices and data relative to weather and terrain surrounding the microwave link. Authors devise a deep learning-based method that achieves a 95%

[1] https://github.com/bonsai-lab-polimi/tnsm2024-asap-venn-abers

classification accuracy. Authors in [23] introduce an anomaly detection system that leverages both the link's performance data (signaling quality and transmission performance) and network topology, while employing active learning techniques to update the detection model continually. The performance data is aggregated and sent to a network management system every 15 minutes, where it is later processed and used for inference. Similarly, in [11], authors introduce an LSTM-based feature fusion network designed to capture both spatial and temporal features within microwave network by incorporating network topology into the LSTM, in addition to links' performance data collected by probes every 15 minutes. Moreover, [12] proposes an ML-based framework that combines eXplainable AI techniques and uncertainty quantification to achieve reliable and robust failure-cause identification. As data, the work utilizes the link's performance measurements aggregated in 15-minute intervals. The work in [24] also examines the problem of failure-cause identification in a scenario involving multiple cooperating operators, i.e., where data is split among operators, and where one operator possesses only partial knowledge of failure causes during the training stage. The authors devised a classification model based on Federated Learning (FL) to train an ML model to detect six distinct failure causes while adhering to privacy constraints.

While these works rely on accumulating telemetry statistics to subsequently perform failure-cause detection, our proposed framework aims to bridge the gap between prediction and telemetry accumulation, where predictions and telemetry accumulation occur concurrently, allowing the ML models to output predictions (in our case, indicating the hardware failure-cause in a microwave link) as soon as they reach a mature stage, referred to as *As-Soon-As-Possible* prediction. By intertwining ML model inference and performance assessment with ongoing telemetry and data collection, our proposed frameworks strive to offer more robust, timely, and actionable failure identification and mitigation.

Finally, the methodological framework proposed in this paper closely resembles ASAP in-network traffic classification in pForest [25], from which we adopt the terminology. Our work brings two major improvements compared to pForest. First, we propose integrating Venn-Abers calibration in the inference phase to provide formal guarantees on the probability of correct classification, while pForest does not provide any such guarantee. Second, we do not make any assumption on the underlying ML model, namely, we do not assume a specific model architecture, nor a minimum level of performance (e.g., classification accuracy) in generalization, while pForest is limited to Random Forests and assumes that the model can deliver a minimum level of performance. Our goal is, therefore, to provide the network manager with an accurate decision-making tool in the form of valid predictive probabilities, regardless of how good or bad the underlying ML model might be.

III. Background on Microwave Networks

In this Section, we provide a brief overview of the main components of a microwave network and introduce our dataset of hardware failures in microwave networks.

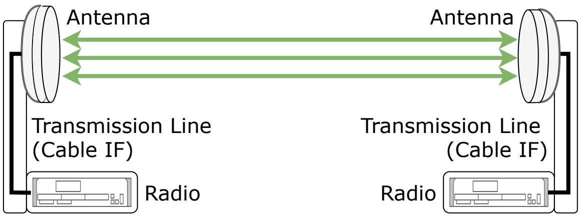
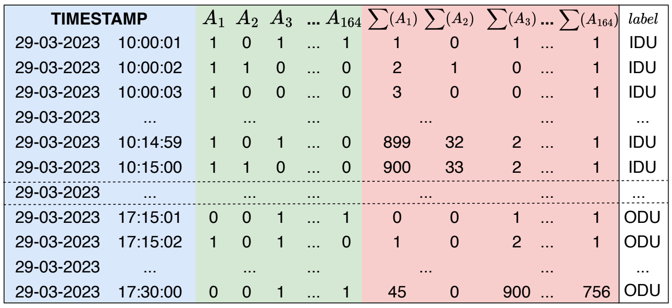


Fig. 2. Basic components of a microwave link.



TIMESTAMP		A_1	A_2	A_3	...	A_{164}	$\sum(A_1)$	$\sum(A_2)$	$\sum(A_3)$	...	$\sum(A_{164})$	*label*
29-03-2023	10:00:01	1	0	1	...	1	1	0	1	...	1	IDU
29-03-2023	10:00:02	1	1	0	...	0	2	1	0	...	1	IDU
29-03-2023	10:00:03	1	0	0	...	0	3	0	0	...	1	IDU
29-03-2023	...		...		...		...			...		...
29-03-2023	10:14:59	1	0	1	...	0	899	32	2	...	1	IDU
29-03-2023	10:15:00	1	1	0	...	0	900	33	2	...	1	IDU
29-03-2023	...		...		...		...			...		...
29-03-2023	17:15:01	0	0	1	...	1	0	0	1	...	1	ODU
29-03-2023	17:15:02	1	0	1	...	0	1	0	2	...	1	ODU
29-03-2023	...		...		...		...			...		...
29-03-2023	17:30:00	0	0	1	...	1	45	0	900	...	756	ODU

Fig. 3. Illustrative representation of the SIAE-Microelettronica hardware failure dataset for one microwave link in two 15-minute windows.

A. Microwave link

Fig. 2 shows the basic structure of a bidirectional microwave link, highlighting the *transmitting end* (TX/RX site) and the *receiving end* (RX/TX site). Each end is composed of three main elements:

1) *Microwave Radio*: generates a signal at the TX/RX site and receives the signal at the RX/TX site. In this work, we consider a split-mount placement, where the electronic devices are distributed between an outdoor unit (ODU) and an indoor unit (IDU). The IDU contains the power unit and electronic components like modems, converters, and the system's interface. The power unit in the IDU provides the necessary electrical power to the entire system, including the ODU. The ODU is mounted outside from the antenna and is powered via the cable from the IDU.
2) *Transmission Line*: connects the microwave radio to the directional antenna through a coaxial cable or a waveguide. The Intermediate Frequency cable (IF cable) connects the IDU and the ODU in a split-mount system. The IF cable is responsible for non-negligible signal losses, depending on signal frequency, and may strongly affect the quality of transmission in case of physical medium deterioration.
3) *Microwave Antenna*: is directional, usually parabolic-shaped, and characterized by its gain, size, and directivity functions. In split-mount systems, the antenna is co-located with the ODU.

We focus on hardware failures that can impact microwave equipment of a microwave link.[2] In particular, 1) hardware failure of the IDU unit (IDU failure), 2) hardware failure of the ODU unit (ODU failure), 3) hardware failure of the

[2] From the three components of the microwave link, we consider hardware failures of the microwave radio and the transmission line. However, we do not consider hardware failures of the antenna.

transmission line (IF cable failure), and 4) hardware failure of the power unit (power failure). In the following, we provide a detailed description of the real-world hardware failure dataset used in our study.

B. Microwave hardware failure bit-sequence dataset

The unavailability of a microwave link is defined in ITU-T Recommendations G.826 and G.828 [26] in terms of *Unavailability Seconds* (UAS), i.e., the number of seconds over which the number of errored bits exceeds a given threshold. UAS may be caused by several phenomena, such as propagation failures due to e.g., atmospheric fading, or hardware failures due to equipment malfunction.

We leverage a real-world dataset of hardware failures from 108 microwave links from a microwave network provided by SIAE Microelettronica [27]. Alarms from each microwave link are collected at *one second* granularity in windows of 15 minutes, with a binary indicator "1" if an alarm signal is ON and a binary indicator "0" if an alarm signal is OFF. Hence, constructing the *alarm bit-sequence dataset*. In addition, from the *alarm bit sequence dataset*, we construct a *15-minute window dataset* by computing the number of seconds an alarm signal is ON in non-overlapping 15-minute windows. For each alarm signal, in a 15-minute window, we get a number ranging between 0 and 900, representing the number of seconds the alarm signal was ON during the 15-minute window.[3]

Our dataset corresponds to 861 disjoint 15-minute window observations of 164 alarm signals collected from 108 microwave links during a time period across two years. Alarm signals are triggered based on the output of multiple telemetry sources installed in the hardware equipment, such as temperature and power sensors, or status monitors for multiple hardware and software sub-components.

The dataset comprises four hardware failure classes: 1) *IDU failure* (e.g., failure of some electronic IDU component, or a temperature issue due to improper equipment installation and/or a worn fan), 2) *ODU failure* (similar to IDU), 3) *Cable failure* (e.g., damaged connectors), and 4) *Power failure* (e.g., due a power outage and/or a battery problem). Each failure type is identified based on alarms issued by the radio equipment, serving as input features to the ML-based classifier. The frequency of each failure class in each 15-minute window is as follows: 1) IDU failure: 129 observations, 2) ODU failure: 493 observations, 3) Cable failure: 75 observations, 4) Power failure: 164 observations.

Fig. 3 shows an illustrative example of the hardware failure dataset for one microwave link. For each link, we consider a *one-second* granularity timestamp observation for each of the 164 alarm signals (A_1 to A_{164}) in 15-minute non-overlapping windows. Additionally, we report the cumulative sum of the number of seconds an alarm signal is ON during the 15-minute window, for each alarm signal ($\sum(A_1)$ to $\sum(A_{164})$). Finally, we report the ground truth *label* of the hardware failure-cause.

We first reconstruct the *bit-sequence dataset* measured every second within the 15-minute telemetry collection window. Then, we construct expanding-window features starting from the beginning of each 15-minute window. Each expanding-window feature represents the number of seconds the corresponding alarm signal was ON at time $t = 1, \ldots, 900$ within the 15-minute window. We leverage this dataset to simulate a streaming scenario where telemetry data is aggregated and fed to an ML model for failure-cause classification in real-time. Note that the 15-minute dataset is a subset of this new dataset. In summary, two datasets are used in our study are:

1) *Expanding window dataset* for the As-Soon-As-Possible prediction. Each observation represents the total number of seconds the alarm signals were ON at a certain time within the 15-minute observation window.
2) *15-minute window dataset* for Aggregate prediction. Each observation aggregates bit sequences in non-overlapping windows of 15 minutes, counting for each window the number of seconds each alarm signal is ON.

IV. As-Soon-As-Possible Failure-Cause Identification in Microwave Networks

We now focus on the problem of failure-cause classification through an ML model, given a stream of microwave equipment alarms. While in this paper we utilize a microwave hardware failure dataset, our methodology can be applied to any dataset of streamed measurements, such as the ones employed in prior work on classifying propagation failures [12]. We first present our proposed As-Soon-As-Possible classification framework, and then discuss its methodological aspects in detail.

A. Reference scenario and problem statement

Our reference scenario is as follows: when a non-zero UAS is detected in a radio link, the alarm bit sequences are streamed to an ML model for inference. In particular, as the stream progresses, we accumulate statistics on the alarm status to construct more accurate representations of the alarm. In our case, we consider computing the total number of seconds each alarm signal is ON. The bit sequences are then fed to an ML model trained for failure-cause classification.

Fig. 4 illustrates and compares a high-level overview of two ML-based methodologies for failure-cause classification, namely, (a) Aggregate prediction, which is the current state-of-the-art, and (b) As-Soon-As-Possible prediction.

Aggregate prediction (Fig. 4, top) consists of accumulating telemetry data for a fixed-size window (in our case, 15 minutes). For failure management in microwave networks, the above methodology displayed remarkable accuracy (above 95% in past literature [21]). However, there are two main drawbacks to this approach. First, aggregate failure classification always requires 15-minute windows, irrespective of the "difficulty" in classifying an observation. Since choosing a mitigation strategy depends on proper failure-cause classification, the aggregate window strategy introduces an unnecessary bottleneck. Second, and more critical, there are

[3] A value "0" means the alarm signal is OFF during the whole window, while a value "900" means the alarm signal is ON during the whole window duration. Any number x: $0 < x < 900$, means the alarm signal is ON for x seconds during the whole window duration. However, this does not imply the alarm signal is ON for x seconds sequentially, as, in practice, an alarm signal may be ON and OFF in different sections of the 15-minute window.

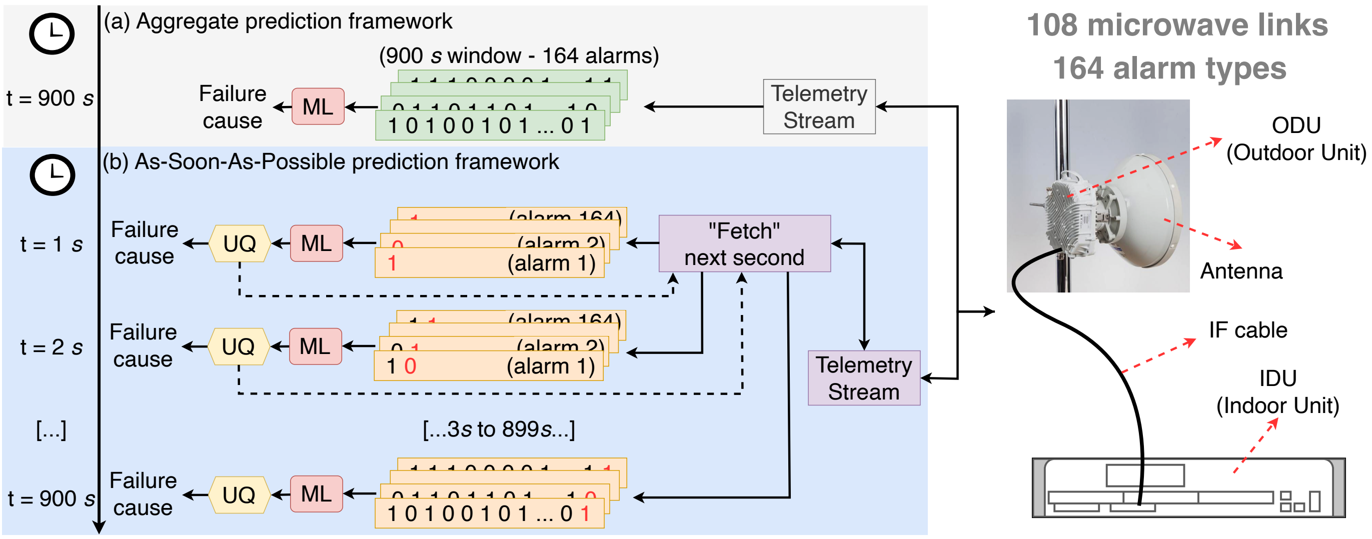


Fig. 4. Aggregate vs. As-Soon-As-Possible (ASAP) failure-cause identification in microwave networks. The aggregate approach collects telemetry data for a fixed-size temporal window (e.g., 15 minutes) and then feeds the extracted features to an ML model for classification. ASAP prediction feeds to the ML model expanding subwindows of telemetry data as soon as they are collected and leverages Uncertainty Quantification (UQ) to return a prediction only when the probability of correct classification is greater than a user-specified safety threshold.

no formal guarantees of the correctness of the prediction. Even if the model displays a test accuracy of 95% (i.e., it performs generally correct class assignment), we have no general guarantees on the probability of correct predictions on new individual samples. In other words, the aggregate method does not quantify how *reliable* the model's predictions are.

To solve the above problems, we propose leveraging *As-Soon-As-Possible* (ASAP) prediction (Fig. 4, bottom). In the ASAP prediction framework, alarms are fed to the ML model as soon as they are collected. By leveraging principled uncertainty quantification via Venn-Abers predictors, the ML classifier will produce a prediction as soon as the probability of correct classification is greater or equal to a user-specified threshold. The advantages of ASAP prediction over aggregate prediction lie in reducing the average time-to-predict (hence, the time for deploying the appropriate mitigation strategy) without compromising on the predictive accuracy. Moreover, by providing a soft probability instead of a hard class assignment, we allow the network operator to make informed decisions based on predictive uncertainty.

Referring to Fig. 4 (b), *As-Soon-As-Possible prediction* shows an example of accumulating the number of seconds each alarm is ON to make a classification decision with high confidence. At a timestamp t (e.g., t = 2s), the ML classifier utilizes the information from *t-1* previous timestamps (e.g., t = 1s). In line with this, we aim to deploy a model that returns accurate predictions as soon as possible. This problem can be formally stated as *selective classification*: the model should return a prediction only if the probability of correct classification is greater or equal to a user-specified safety threshold $1 - \alpha$; otherwise, it will abstain. In other words, the classifier should differentiate between easy-to-classify and hard-to-classify hardware failures and decide accordingly. For example, predictions for easy-to-classify failures may be returned at the first second (t = 1s). In contrast, for hard-to-classify failures, the model might abstain from predicting until several hundreds of seconds of alarms have been observed (e.g., t = 900s in case of the complete 15-minute window).

We remark that, when a user enforces a safety threshold (e.g., 95% probability of correct classification), the model may not guarantee that level of certainty for every example at the end of the 900s monitoring window. In other words, in the case of particularly hard-to-classify examples, the model will abstain from predicting. We refer to these samples as *rejected* samples. Rejected samples might be, for instance, sent to domain experts for a more detailed inspection [12]. The rejection rate ultimately depends on the predictive power of the ML model: a highly accurate model implies lower rejection rates, and vice-versa. In this context, we emphasize that our framework enforces safety constraints without assumptions about the underlying ML model's performance.

In the following discussion, we assume that the ML classifier is a *scoring classifier* that can output a heuristic measure of confidence on its predictions, e.g., in the form of normalized scores in $[0, 1]$ for each output class. Many popular ML models are scoring classifiers: for instance, artificial neural networks output softmax probabilities, while decision trees output the class frequency in the leaf nodes. From this assumption, we discuss two different strategies for implementing ASAP failure prediction. First, we discuss the baseline strategy of thresholding the class scores, which is a straightforward extension of the current state-of-the-art, and we highlight its fundamental limitations. We then propose leveraging Venn-Abers predictors to overcome these limitations.

B. *ASAP failure classification via score thresholding*

A straightforward solution for ASAP failure classification is to threshold the predicted class scores, returning a prediction only if the score associated with the most likely class is greater than $1 - \alpha$. Formally, at each time-instant $t \in [1, 900]$ in the alarm collection window, we return a class prediction $\hat{y}_t$ given features $\mathbf{x}_t$ as follows:

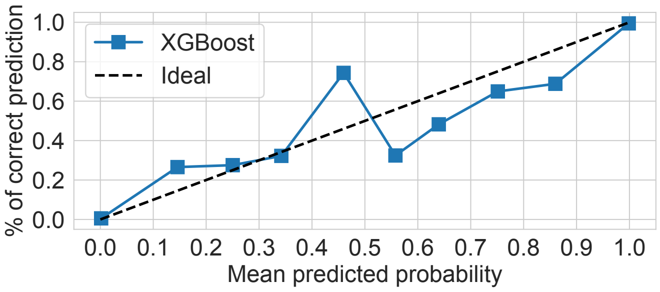


Fig. 5. Reliability diagram of an XGBoost model on the hardware failures dataset. The XGBoost model is overconfident for high-confidence predictions.

$$\hat{y}_t = \begin{cases} \arg\max \hat{f}(\mathbf{x}_t) & \text{if } \max \hat{f}(\mathbf{x}_t) \geq 1-\alpha, \\ \emptyset & \text{otherwise,} \end{cases} \tag{1}$$

where $\emptyset$ stands for "no prediction". Unfortunately, the classification scores produced by an ML model are generally not valid probabilities in the statistical sense [28], [29]. Formally, we say that a model is *calibrated* if [29]:

$$p(c_j \mid p^{c_j}) = p^{c_j} \quad \forall p^{c_j} \in [0,1], \tag{2}$$

where p^{c_j} is the predicted probability for class c_j, and $p(c_j \mid p^{c_j})$ is the probability of the ground-truth being c_j if the corresponding predicted probability is p^{c_j}. For example, in a perfectly calibrated model, a class prediction with a score equal to 0.95 has a 95% chance of being correct. Generally, a trained ML model is not calibrated out of the box [28], [29]. As an illustrative example, Fig. 5 shows the *reliability diagram* of an XGBoost classifier in an 80-20 train-test split of our 15-minute window dataset. This diagram displays the mean predicted probability versus the empirical frequency of correct classification in ten equally-spaced bins between $[0,1]$. An ideal, perfectly calibrated model would display points located on the diagonal of the diagram. In our case, we observe that the XGBoost model is, on average, not well-calibrated. For example, for a mean predicted probability of 86%, the actual frequency of correct classifications is only 69%. We conclude that, though the model is very skilled at *class assignment* ($> 95\%$ test set accuracy), it is *overconfident* when outputting high-confidence predictions. In other words, the high predictive power of the current state-of-the-art does not guarantee probability calibration, which is highly undesirable for high-risk failure management applications. Though this illustrative example considers one specific split and model class, in general, we have no control on the default calibration of an ML model. We, therefore, need more sophisticated methodologies for producing reliable, high-confidence predictions that do not assume the model to be calibrated by default.

We consider the problem of *probability calibration*, that is, how to transform heuristic uncertainty estimates of an ML model into valid probabilities. Popular classical approaches for probability calibration are Platt scaling [30] and isotonic regression [31]. Unfortunately, the correctness of these approaches depends on assumptions that are unrealistic in many practical scenarios. In particular, Platt scaling assumes that the reliability diagram of the ML classifier to be calibrated has a sigmoidal shape, while isotonic regression assumes a perfectly monotonic relationship between the predicted scores and the true probabilities. In summary, classical approaches like Platt scaling and isotonic regression can be regarded as heuristics with no formal validity guarantees.

C. ASAP failure classification via Venn-Abers predictors

An alternative approach emerging as a promising solution to overcome the shortcomings of the aforementioned approaches are Venn-Abers predictors (VAPs) [9], [10]. VAPs are a methodological framework for turning heuristic probability estimates from an arbitrary ML model into rigorous probability estimates with theoretical guarantees on calibration. The only assumption is the availability of a *calibration dataset* $\mathcal{D}_{\text{cal}} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n_{\text{cal}}}$, where $\mathbf{x}$ and y indicate features and ground-truth annotations, respectively. The calibration set represents "fresh" i.i.d. data not used during training. VAPs leverage the ML model's predictive performance on the calibration set to turn class scores into well-calibrated probabilities. VAPs are, in principle, defined for binary classification problems. We first briefly discuss the algorithm for the binary classification case and then extend it to the multiclass case.

Binary VAPs. We consider two variants of VAPs, namely Inductive Venn-Abers predictors (IVAPs) and Cross Venn-Abers (CVAPs) predictors. IVAPs provide formal theoretical guarantees on calibration. CVAPs are derived from IVAPs, exhibiting stronger empirical performance on average while dropping theoretical guarantees on calibration.

It can be demonstrated that, unfortunately, it is impossible to learn an optimal probabilistic classifier from a finite-size dataset [32], [33]. VAPs overcome this challenge by a) producing two probabilities instead of one, and b) restricting the optimality guarentees to calibration.

Formally, VAPs are *multiprobabilistic predictors*, that is, for each test example, they produce two probabilities (p_0, p_1) for the sample to be assigned to the positive class. VAPs guarantee that either p_0 or p_1 will be perfectly calibrated, according to the definition in Eq. (2). The key intuition is as follows: if p_0 and p_1 are close to each other, and one of them is calibrated, then we can expect that their "average" will be also calibrated.

Indeed, for decision-making purposes, we need one single probability value. Unfortunately, it is generally impossible to know which one among p_0 or p_1 is the "right" choice. To solve this problem, p_0 and p_1 are merged into a single value p that minimizes the error with respect to a proper scoring rule, e.g., the log-loss or the Brier score. We now discuss each of the above steps in more detail.

Algorithm 1 and Algorithm 2 illustrate the procedure for building IVAPs and CVAPs, respectively. IVAPs split the training set D_{train} into a "proper" training set D'_{train} and a calibration set D_{cal}. First, we fit the ML classifier to the proper training set. Then, we fit two Isotonic Regression algorithms (f_0 and f_1 in Algorithm 1) to the prediction scores in the calibration set and the predicted score on the test sample. IVAPs output two probability values, each one assuming that the ground-truth value for the test sample is either 0 or 1. The intuition is that, since the ground truth for the test sample is either 0 or 1, one of the two probabilities will be calibrated.

Algorithm 1 Inductive Venn-Abers Predictor (IVAP)

```
Require: Scoring ML classifier $f$ (e.g., XGBoost), proper training dataset $\mathcal{D}'_{\text{train}} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n_{\text{train}}}$, calibration dataset $\mathcal{D}_{\text{cal}} = \{(\mathbf{x}_j, y_j)\}_{j=1}^{n_{\text{cal}}}$, test example $\mathbf{x}_{\text{test}}$
1: $\hat{f} \leftarrow f$.fit($\mathcal{D}_{\text{train}}$)
2: $\{s_i\}_{i=1}^{n_{\text{cal}}} \leftarrow \{\hat{f}(\mathbf{x}_j)\}_{i=1}^{n_{\text{cal}}}$    // compute calibration scores
3: $s_{\text{test}} \leftarrow \hat{f}(\mathbf{x}_{\text{test}})$    // compute test score
4: $f_0 \leftarrow$ Isotonic.fit($(s_0, y_0), \dots, (s_{n_{\text{cal}}}, y_{n_{\text{cal}}}), (s_{\text{test}}, 0)$)
5: $f_1 \leftarrow$ Isotonic.fit($(s_0, y_0), \dots, (s_{n_{\text{cal}}}, y_{n_{\text{cal}}}), (s_{\text{test}}, 1)$)
6: $(p_0, p_1) \leftarrow (f_0(s_{\text{test}}), f_1(s_{\text{test}}))$
7: return $(p_0, p_1)$
```

Algorithm 2 Cross Venn-Abers Predictor (CVAP)

```
Require: Scoring ML classifier $f$, training dataset $\mathcal{D}_{\text{train}} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n_{\text{train}}}$, test example $\mathbf{x}_{\text{test}}$, number of folds $k$
Split the training set $\mathcal{D}_{\text{train}}$ into $k$ folds $\mathcal{D}_1, \dots, \mathcal{D}_k$
for $i \leftarrow 1$ to $k$ do
    $(p_0^i, p_1^i) \leftarrow$ IVAP($f, \mathcal{D}_{\text{train}} \setminus \mathcal{D}_i, \mathcal{D}_i, x_{test}$)
end for
return gmean($\mathbf{p}_1$) / (gmean($1 - \mathbf{p}_0$) + gmean($\mathbf{p}_1$))
```

Note that p_0 and p_1 are not complementary, i.e., $p_1 \neq 1 - p_0$. In fact, it always holds that $p_0 < p_1$. In practice, for reasonably-sized datasets, p_0 and p_1 will have similar values [10]. Because of this, if one of the two is perfectly calibrated, we can expect that their "average" will be well-calibrated. As such, we merge the two probabilities p_0 and p_1 into a single value, as follows:

$$p = \frac{p_1}{1 - p_0 + p_1}. \tag{3}$$

Choosing this value of p yields log-minimax IVAPs, that is, it minimizes the regret of using p instead of the appropriate p_0 or p_1 over the log-loss. We point the reader to a complete proof of the calibration of IVAPs [34] and the log-minimax rule for deriving p [9].

CVAPs are an extension of IVAPs, dropping the theoretical guarantees on calibration in favor of a stronger empirical performance [9]. The algorithm splits the training set into k non-overlapping folds and applies IVAP k times, each time considering one fold as the calibration set and the remainder folds as the proper training set. This procedure results in two vectors $(\mathbf{p}_0, \mathbf{p}_1)$, where (p_0^i, p_1^i) are the outputs of IVAP considering the i-th fold as calibration set. We then merge the vectors into a single probability value, as follows:

$$p = \frac{\text{gmean}(\mathbf{p}_1)}{\text{gmean}(1 - \mathbf{p}_0) + \text{gmean}(\mathbf{p}_1)}, \tag{4}$$

where gmean$(\cdot)$ indicates the geometric mean. Note that, Eq. (3) is a special case of Eq. (4) when $k = 1$. As for Eq. (3), it can be demonstrated that choosing p as in Eq. (4) minimizes the regret over the log-loss, i.e., is log-minimax [9].

Multiclass VAPs. So far, we have discussed VAPs in the binary classification case. We now outline different techniques for generalizing these algorithms to the multiclass case.

A first approach consists of treating the multiclass problem into multiple one-versus-all (binary) classification problems, and constructing a VAP for each class [35]. However, in our specific application scenario (selective classification), we are not interested in calibrating *every* predicted probability, but only the probability of the predicted class. To this end, we follow the guidelines of prior work [36] and apply VAPs only for calibrating the probability that the predicted class is correct. To do so, we relabel each instance in the calibration set to 1 if the ground-truth class is equal to the predicted class of the model, and 0 otherwise. Formally, with reference to Algorithms 1 and 2, after splitting the data in training and calibration set, we construct a new calibration set $\mathcal{D}_{\text{cal}}^{\text{bin}} = \{(\mathbf{x}_i, y_i^{\text{bin}})\}_{i=1}^{n_{\text{cal}}}$ with binary labels, as follows:

$$y_i^{\text{bin}} = \begin{cases} 1 & \text{if } y_i = \arg\max \hat{f}(\mathbf{x}_i), \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

The remainder of the IVAP and CVAP algorithms can be executed without further modifications. The final result is a VAP that calibrates the probability of correct classification, which can then be leveraged for ASAP classification.

Computational complexity of VAPs: we remark that building IVAPs and CVAPs as per Algorithms 1 and 2 appears computationally onerous, as it requires fitting two isotonic regression model for each test example. However, we note that the two isotonic regression models are fitted to the calibration scores plus only one other example, namely, the test example. One can leverage this property to reduce the computational complexity of the calibration procedure (lines 4-6 of Algorithm 1) to $O(n_{\text{cal}} \log(n_{\text{cal}}))$, where n_{cal} is the number of examples in the calibration set. The remainder of the total complexity is dominated by model training (line 1 of Algorithm 1), which is performed only once and in an offline phase. This makes the application of VAPs feasible for our application scenario, where the ML model must supply a prediction every second. Deriving the efficient algorithm is non-trivial, therefore, we omit the proof for brevity. We refer the readers to a detailed derivation [9] and a reference Python implementation of efficient IVAPs [37].

V. Illustrative Numerical Results

We now discuss illustrative numerical results highlighting the practical advantages brought by our ASAP classifier compared to the state-of-the-art. In particular, we empirically show that our ASAP classifier can:

1) Satisfy (on average over 10-fold cross-validation splits) a user-specified high probability of correct classification (95% or 99%), while rejecting a reasonable number of test samples (on average less than 10% for achieving > 95% selective accuracy)
2) Achieve ~8x speedups on the mean time-to-predict compared to a state-of-the-art failure-cause classification aggregate strategy operating on 15-minute windows, while maintaining the same level of accuracy (> 95%)

We consider a single XGBoost model with default configuration as an ML classifier [13]. This is because gradient-boosted tree models are the current state-of-the-art in ML for tabular data [38], [39]. We borrow from *VennABERS.py* [37] for implementing IVAPs and CVAPs. Results are aggregated

TABLE I

PROBABILISTIC PREDICTION METRICS FOR UNCALIBRATED, IVAP AND CVAP CLASSIFIERS. WE REPORT MEAN AND STANDARD DEVIATIONS OVER 10-FOLD CROSS-VALIDATION.

Metric	Uncalibrated	IVAP	CVAP
Log-loss	0.21 ± 0.06	0.14 ± 0.04	0.13 ± 0.03
Brier score	0.09 ± 0.02	0.04 ± 0.01	0.039 ± 0.009
ECE	0.028 ± 0.015	0.025 ± 0.012	0.023 ± 0.009

over 10-fold cross-validation. To avoid data leakage, we perform train-test splits over disjoint 15-minute windows.

As a baseline algorithm, we consider an Uncalibrated ASAP classifier trained on the expanding-window dataset that thresholds the XGBoost class scores without applying any post-processing (Section IV-B). Note that, while CVAP and IVAP require holding out a portion of the training set for calibration, we train the uncalibrated classifier on the whole training set for a fair comparison. For IVAP, we hold out 20% of the training dataset for calibration. For CVAP, we consider splitting the training dataset into $k = 5$ folds.

We first quantitatively evaluate whether or not IVAPs and CVAPs provide better probabilistic predictions compared to the Uncalibrated model. We report three metrics for probabilistic prediction: i) log-loss, ii) Brier score, and iii) Expected Calibration Error (ECE). The log-loss is defined as follows:

$$\text{Log-loss} = \begin{cases} -\log(p) & \text{if correct,} \\ -\log(1-p) & \text{otherwise,} \end{cases} \tag{6}$$

where p is the probability associated to the predicted class. The Brier score is defined as follows:

$$\text{Brier score} = \begin{cases} (p-1)^2 & \text{if correct,} \\ ((1-p)-1)^2 & \text{otherwise.} \end{cases} \tag{7}$$

The Brier score can be interpreted as the mean squared error applied to predicted probabilities. We remark that both the log-loss and the Brier score are *strictly proper scoring rules* [40], that is, lower scores indicate a better approximation of the true data distribution. Finally, ECE is defined as the weighted average of the absolute difference between the mean of the predicted probabilities (mop) and the fraction of correct predictions (foc) in $M = 10$ equally-spaced bins (the same computation that resulted in Fig. 5), as follows:

$$\text{ECE} = \sum_{i=1}^{M} \frac{|B_i|}{n_{\text{test}}} |\text{foc}(B_i) - \text{mop}(B_i)|. \tag{8}$$

We underline that, unlike the log-loss and the Brier score, the ECE is not a proper scoring rule, that is, lower ECE *does not* generally imply better probabilistic predictions. As a simple counterexample, a classifier always predicting the class frequencies in the training set will achieve near-zero ECE, despite being useless for making predictions. Still, we report the metric for its useful intuitive interpretation.

Table I illustrates the average log-loss, Brier score, and ECE of the Uncalibrated XGBoost model, IVAP, and CVAP. We observe that VAPs achieve better values for all the considered metrics. In particular, since VAPs achieve better log-loss and Brier score, we can conclude that they can provide better probabilistic predictions than the Uncalibrated model.

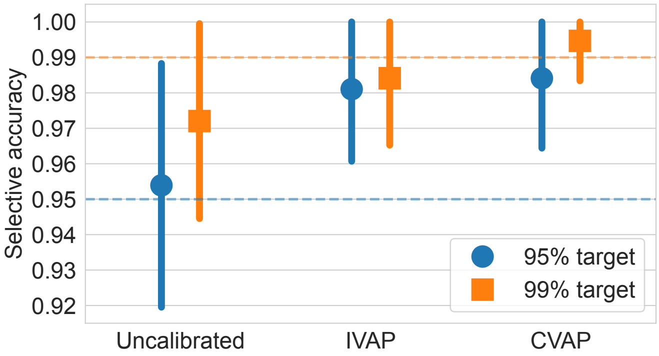


Fig. 6. Selective accuracy of Uncalibrated, IVAP and CVAP for thresholds 95%, 99% on the probability of correct classification. We report means and standard deviations over 10-fold cross-validation.

TABLE II

SELECTIVE ACCURACY AND REJECTION RATIOS OF UNCALIBRATED, IVAP AND CVAP FOR ASAP CLASSIFICATION. WE REPORT MEAN AND STANDARD DEVIATIONS OVER 10-FOLD CROSS-VALIDATION.

Metric	Uncalibrated	IVAP	CVAP
		Target: > 0.95 selective accuracy	
Selective accuracy	0.954 ± 0.033	0.981 ± 0.019	0.984 ± 0.019
Rejection ratio	0.3% ± 0.5%	5% ± 3%	8% ± 4%
		Target: > 0.99 selective accuracy	
Selective accuracy	0.972 ± 0.026	0.984 ± 0.018	0.994 ± 0.011
Rejection ratio	1.0% ± 0.9%	11% ± 6%	18% ± 6%

We now evaluate our approach in terms of *selective accuracy*, that is, the frequency of correct classification of the accepted predictions. Fig. 6 illustrates the selective accuracy of the ASAP model under error rate constraints of $\alpha = 0.05$ and $\alpha = 0.01$ (i.e., 95% and 99% accuracy, respectively). For the 95% case, we observe that even though the uncalibrated model delivers, on average, a 95% selective accuracy, it does not perform consistently among different splits. Indeed, the minimum selective accuracy is below 91%, illustrating that the uncalibrated model tends to be overconfident. From a failure management perspective, this can result in underestimating the risk associated with accepting wrong failure classifications, which may have a devastating impact if said classifications are leveraged for choosing a mitigation strategy. Conversely, both IVAP and CVAP deliver a selective accuracy always greater than 0.95%. For the 99% case, we observe that both the uncalibrated model and IVAP do not meet the target and deliver an accuracy less than 99%. In contrast, CVAP delivers an average selective accuracy above 99% with relatively low variance across the splits. This is a remarkable result, as our training dataset is relatively small, and we are querying the extreme tail of the predictive distribution. We draw two main conclusions from this first analysis. First, calibrating via either IVAP or CVAP consistently improves the selective accuracy over the uncalibrated model. Second, consistent with prior findings [9], we conclude that although CVAPs drop the theoretical guarantees of IVAPs, they provide better empirical performance. From the point of view of network management, the model's predictions provided by our framework are highly reliable for supporting decision-making.

We complement the comparisons in Fig. 6 with Table II, which reports the percentage of rejected test samples

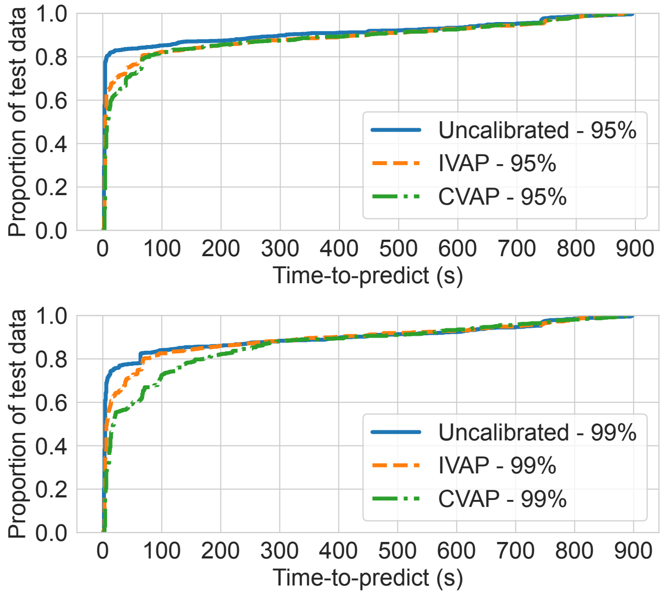


Fig. 7. Time-to-predict CDFs for Uncalibrated, IVAP and CVAP for selective classification under 95%, 99% probability of correct prediction.

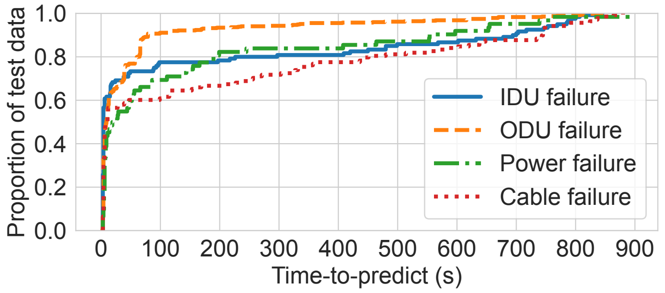


Fig. 8. Per-class time-to-predict CDFs of the CVAP classifier under a 95% constraint on the probability of correct prediction.

for Uncalibrated, IVAP and CVAP for thresholds 95%, and 99% on the probability of correct classification. As expected, the more stringent the threshold is, the greater the rejection ratio. We observe that the uncalibrated model yields very low rejection rates, a few percent on average; however, as anticipated before, this comes at the price of a selective accuracy that is, on average, lower than the safety threshold. In contrast, IVAPs and CVAPs achieve a reasonable rejection ratio (less than 10% and 20% of the total test samples for 95% and 99% selective accuracy, respectively) while providing well-calibrated predictive probabilities. Finally, we comment on why IVAPs and CVAPs yield rejection rates relatively high compared to the uncalibrated model. Recall that the underlying ML models in IVAP and CVAP are trained with 20% less training data than the uncalibrated model, as they require a held-out calibration set. In other words, they have less predictive power than the uncalibrated classifier. As such, decisions must be more conservative to satisfy the selective accuracy constraint, i.e., the model will reject more test samples. Sacrificing predictive power in favor of calibration is a trade-off that needs to be carefully evaluated when leveraging VAPs. We argue that in a high-risk application such as failure management, well-calibrated probabilistic predictions are more useful for decision-making than hard predictions with no uncertainty quantification. We expect rejection rates to diminish in a scenario of relative data abundance (e.g., thousands of observations), where subtracting 20% of the training set for calibration has a negligible impact on the underlying ML model's performance.

We now quantitatively assess the improvements brought by our ASAP framework in terms of time-to-predict. Fig. 7 illustrates the cumulative distribution of the time elapsed before the model outputs an $(1-\alpha)$-confident prediction for $\alpha = 0.05$ and $\alpha = 0.01$. We see that, in general, most predictions are provided already at the first second. This is expected, since a baseline model operating only on one-second telemetry already displayed accuracy close to 90%. Moreover, we observe that the uncalibrated model outputs predictions on average earlier than IVAP and CVAP. This, however, comes at the price of an inferior selective accuracy, as previously discussed for Fig. 6. Conversely, IVAPs and CVAPs can output the majority of predictions in a few minutes, while respecting the safety threshold. In particular, we observe that, for the 99% case, the average time-to-predict increases compared to 95%. This confirms that the model can indeed delay its prediction until it matures a sufficient level of confidence to satisfy the safety threshold. In particular, the CVAP classifier yields an average time-to-predict equal to $101s$ and $119s$ for 95% and 99% accuracy targets, respectively. This grants a mean speedup of 8.9x and 7.5x, respectively, compared to the state-of-the-art aggregate classifier operating on 15-minute windows. We conclude that our ASAP classifier can provide both reliable and fast predictions.

Fig. 8 breaks down Fig. 7 on each individual failure class for the CVAP classifier under a 95% selective accuracy constraint. Note that, in this case, the y-axis reports the proportion of test data per class. We observe that some failure classes are predicted on average later than others. In particular, we observe that, cable failures are predicted on average later than all other failures. This is an expected result, considering the class distribution in our dataset. Recall that the cable failure is the least represented among all the failure classes with only 75 observation (6.6x less than ODU failure, the most represented class). It is a well-known result that, in scenarios with class imbalance, a ML model will tend to favor the better-represented classes [41], [42]. For this reason, we can expect the predictions for the cable class to be, on average, more uncertain compared to the other failure classes, resulting in a longer time-to-predict on average. The information provided by the above analysis can also be leveraged in a counterfactual manner. For example, in case the ML model outputs several consecutive predictions with high uncertainty, regardless the output classes in these predictions, one can conclude that the most likely class is among those that are typically predicted later, e.g., cable failures, according to our analysis.

VI. Conclusion

In this paper, we introduced As-Soon-As-Possible (ASAP) selective classification for hardware-failure-cause identification in microwave networks. In contrast to the current state-of-the-art, which leverages data collected in fixed-size measurement windows, our ASAP classification framework is designed to output a prediction as soon as the probability of correct classification exceeds a user-specified threshold. To this end, we leverage recent advances in the field of Venn-Abers predictors, which allows to turn any scoring classifier into a well-calibrated probabilistic classifier. Overall, our framework empowers the network manager with prompt and reliable failure-cause predictions, reducing the time-to-predict by ∼8x while ensuring a selective accuracy greater than 95%. Future research directions include the investigation of feature importance in determining the selective accuracy of the prediction, i.e., to identify whether using or removing any alarm from the features set has a positive or a negative impact on the time-to-predict.

References

[1] C.-X. Wang *et al.*, "On the road to 6g: Visions, requirements, key technologies, and testbeds," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 2, pp. 905–974, 2023.

[2] M. Banafaa *et al.*, "6g mobile communication technology: Requirements, targets, applications, challenges, advantages, and opportunities," *Alexandria Engineering Journal*, vol. 64, pp. 245–274, 2023.

[3] Z. Qadir, K. N. Le, N. Saeed, and H. S. Munawar, "Towards 6G internet of things: Recent advances, use cases, and open challenges," *ICT Express*, vol. 9, no. 3, pp. 296–312, 2023.

[4] J. Mata *et al.*, "Artificial intelligence (ai) methods in optical networks: A comprehensive survey," *Optical switching and networking*, vol. 28, pp. 43–57, 2018.

[5] M. F. Silva, A. Pacini, A. Sgambelluri, and L. Valcarenghi, "Learning long-and short-term temporal patterns for ml-driven fault management in optical communication networks," *IEEE Transactions on Network and Service Management*, vol. 19, no. 3, pp. 2195–2206, 2022.

[6] L. Gupta *et al.*, "Fault and performance management in multi-cloud virtual network services using ai: A tutorial and a case study," *Computer Networks*, vol. 165, p. 106950, 2019.

[7] J. Gallego-Madrid *et al.*, "Machine learning-based zero-touch network and service management: A survey," *Digital Communications and Networks*, vol. 8, no. 2, pp. 105–123, 2022.

[8] D. Wang *et al.*, "A review of machine learning-based failure management in optical networks," *Science China Information Sciences*, vol. 65, no. 11, p. 211302, 2022.

[9] V. Vovk, I. Petej, and V. Fedorova, "Large-scale probabilistic predictors with and without guarantees of validity," in *Advances in Neural Information Processing Systems*, vol. 28, 2015.

[10] V. Vovk, A. Gammerman, and G. Shafer, *Algorithmic Learning in a Random World*. Berlin, Heidelberg: Springer-Verlag, 2005.

[11] Z. Ruan *et al.*, "Microwave link failures prediction via lstm-based feature fusion network," in *2021 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2021, pp. 1–8.

[12] O. Ayoub *et al.*, "Explainable artificial intelligence in communication networks: A use case for failure identification in microwave networks," *Computer Networks*, vol. 219, p. 109466, 2022.

[13] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, p. 785–794.

[14] F. Lateano, O. Ayoub, F. Musumeci, and M. Tornatore, "Machine-learning-assisted failure prediction in microwave networks based on equipment alarms," in *2023 19th International Conference on the Design of Reliable Communication Networks (DRCN)*. IEEE, 2023, pp. 1–7.

[15] M. Nouioua *et al.*, "A survey of machine learning for network fault management," *Machine Learning and Data Mining for Emerging Trend in Cyber Dynamics: Theories and Applications*, pp. 1–27, 2021.

[16] 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.

[17] R. Gu, Z. Yang, and Y. Ji, "Machine learning for intelligent optical networks: A comprehensive survey," *Journal of Network and Computer Applications*, vol. 157, p. 102576, 2020.

[18] M. Bubniak, P. Musil, and P. Mlŷnek, "Application for an early detection of transmission parameters degradation of point-to-point microwave links," in *2020 12th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*. IEEE, 2020, pp. 82–86.

[19] D. Jacoby, J. Ostrometzky, and H. Messer, "Short-term prediction of the attenuation in a commercial microwave link using lstm-based rnn," in *2020 28th European Signal Processing Conference (EUSIPCO)*. IEEE, 2021, pp. 1628–1632.

[20] ——, "Model-based vs. data-driven approaches for predicting rain-induced attenuation in commercial microwave links: A comparative empirical study," in *2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–5.

[21] F. Musumeci *et al.*, "Supervised and semi-supervised learning for failure identification in microwave networks," *IEEE Transactions on Network and Service Management*, vol. 18, no. 2, pp. 1934–1945, 2020.

[22] M. Choi, T. Kim, J. pil Lee, and S. Koh, "An empirical study on root cause analysis and prediction of network failure using deep learning," in *2021 International Conference on Information and Communication Technology Convergence (ICTC)*. IEEE, 2021, pp. 741–746.

[23] L. Pan *et al.*, "Proactive microwave link anomaly detection in cellular data networks," *Computer Networks*, vol. 167, p. 106969, 2020.

[24] T. Tandel, O. Ayoub, F. Musumecil, C. Passera, and M. Tornatore, "Federated-learning-assisted failure-cause identification in microwave networks," in *2022 12th International Workshop on Resilient Networks Design and Modeling (RNDM)*. IEEE, 2022, pp. 1–7.

[25] C. Busse-Grawitz, R. Meier, A. Dietmüller, T. Bühler, and L. Vanbever, "pforest: In-network inference with random forests," 2022.

[26] Understanding ITU-T error performance recommendations. hhttps://www.julesbartow.com/Pictures/ITS/ITU-T_Errors_ApplicationNote2.pdf.

[27] SIAE microwave product portfolio. https://www.siaemic.com/index.php/products-services/telecommunication-systems/microwave-product-portfolio.

[28] A. Niculescu-Mizil and R. Caruana, "Predicting good probabilities with supervised learning," in *Proceedings of the 22nd International Conference on Machine Learning*, 2005, p. 625–632.

[29] M. Minderer *et al.*, "Revisiting the calibration of modern neural networks," in *Advances in Neural Information Processing Systems*, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., 2021.

[30] J. Platt, "Probabilistic outputs for support vector machines and comparison to regularized likelihood methods," in *Advances in Large Margin Classifiers*, vol. 10, no. 3, 1999, pp. 61 – 74.

[31] M. Ayer *et al.*, "An Empirical Distribution Function for Sampling with Incomplete Information," *The Annals of Mathematical Statistics*, vol. 26, no. 4, pp. 641 – 647, 1955.

[32] A. Gammerman, V. Vovk, and V. Vapnik, "Learning by transduction," in *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, ser. UAI'98. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1998, p. 148–155.

[33] J. Lei and L. Wasserman, "Distribution-free Prediction Bands for Non-parametric Regression," *Journal of the Royal Statistical Society Series B: Statistical Methodology*, vol. 76, no. 1, pp. 71–96, 07 2013.

[34] I. Nouretdinov *et al.*, "Inductive Venn-Abers predictive distribution," in *Proceedings of the Seventh Workshop on Conformal and Probabilistic Prediction and Applications*, vol. 91, 11–13 Jun 2018, pp. 15–36.

[35] V. Manokhin, "Multi-class probabilistic classification using inductive and cross Venn–Abers predictors," in *Proceedings of the Sixth Workshop on Conformal and Probabilistic Prediction and Applications*, vol. 60, 2017, pp. 228–240.

[36] U. Johansson *et al.*, "Calibrating multi-class models," in *Proceedings of the Tenth Symposium on Conformal and Probabilistic Prediction and Applications*, vol. 152, 08–10 Sep 2021, pp. 111–130.

[37] P. Toccaceli, "VennABERS.py," https://github.com/ptocca/VennABERS, 2022, [Online; accessed 7-Dec-2022].

[38] R. Shwartz-Ziv *et al.*, "Tabular data: Deep learning is not all you need," in *8th ICML Workshop on Automated Machine Learning*, 2021.

[39] L. Grinsztajn, E. Oyallon, and G. Varoquaux, "Why do tree-based models still outperform deep learning on typical tabular data?" in *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022.

[40] T. Gneiting and A. E. Raftery, "Strictly proper scoring rules, prediction, and estimation," *Journal of the American statistical Association*, vol. 102, no. 477, pp. 359–378, 2007.

[41] G. Lemaître *et al.*, "Imbalanced-learn: a python toolbox to tackle the curse of imbalanced datasets in machine learning," *Journal of Machine Learning Research*, vol. 18, no. 1, p. 559–563, jan 2017.

[42] M. A. M. Mateusz Buda, Atsuto Maki, "A systematic study of the class imbalance problem in convolutional neural networks," *Neural Networks*, vol. 106, pp. 249–259, 2018.