Microcontroller Performance Screening: Optimizing the Characterization in the Presence of Anomalous and Noisy Data

Microcontroller Performance Screening: Optimizing the Characterization in the Presence of Anomalous and Noisy Data

Nicolò Bellarmino*, Riccardo Cantoro*, Martin Huch†, Tobias Kilian†‡,
Ulf Schlichtmann‡ and Giovanni Squillero*

*Politecnico di Torino Torino, Italy
†Infineon Technologies AG Munich, Germany
‡Technical University of Munich Munich, Germany

***Abstract*—In safety-critical applications, microcontrollers must satisfy strict quality constraints and performances in terms of $F_{\max}$, that is, the maximum operating frequency. It has been demonstrated that data extracted from on-chip speed monitors can model the $F_{\max}$ of integrated circuits by means of machine learning models, and that those models are suitable for the performance screening process. However, while acquiring data from these monitors is quite an accurate process, the labelling is time-consuming, costly, and may be subject to different measurements errors, impairing the final quality. This paper presents a methodology to cope with anomalous and noisy data in the context of the multi-label regression problem of microcontroller performance screening. We used outlier detection based on Inter Quartile Range (IQR) and Zscore and imputation techniques to detect errors in the labels and to avoid to drop incomplete samples, building higher-quality training set for our models, optimizing the devices characterization phase. Experiments showed that the proposed methodology increases the performance of existing models, making them more robust. These techniques permitted us to use a significantly smaller number of samples (about one third of the devices available for characterization) , thus making the costly data acquisition process more efficient.**

***Index Terms*—Performance Screening, Fmax, Speed Monitors, Machine Learning, Active Learning, Device Testing**

I. Introduction

The goal of Microcontroller (MCU) performance screening is to detect under-performing devices that do not fully satisfy the characteristics described in the datashet in terms of maximum operating frequency. To do this, one possibility is to use machine learning (ML) models trained on data that can be correlated to the speed $F_{\max}$ of the devices. The model is derived from available data, and it is able to express the relation between inputs (*features*) and outputs (*labels)*. The dataset creation requires the acquisition of features and labels, and the performance of supervised ML models depends on the quality and the quantity of labeled data. Previous works have proposed the usage of Speed Monitors (SMONs) to predict $F_{\max}$ values derived from the execution of functional patterns on single devices [1]–[4]. In this context, measurements of frequency are prone to errors, dirty data and missing values. Also, the amount of available data is limited because the process of acquiring the labels is costly in terms of time required.

Authors are listed in alphabetical order.

All these problems reduce the quality of the data used in the training procedure and affect the performances of ML models. The label acquisition is performed by humans. Significant problems with manual labelling include the introduction of bias in the data, the huge cost (in terms of time and money, and also the need for a domain-expert that needs to do the work), and the absence of a real "gold standard" for the labels: every manual data acquisition introduces a possible error in the dataset that may be difficult to catch and identify. In addition, the possible presence of noise in the data may make the available data deviates from real values, introducing a shift in the latent data distribution. Noise and many other causes create *anomalous data*, i.e., data that are clearly abnormal with respect to the other entries of our dataset.

Outlier detection techniques can be used to detect and, where appropriate, remove anomalous observations from data. But the majority of machine learning models are not able to deal with *missing values* (especially in the labels). In a classical ML framework, if a particular label has a missing value, the the data point is dropped. But this operation would lead to a smaller dataset, and since the amount of data is limited, to a possible loss of precious information. Several works can be found in the ML literature applied to MCU performance screening ([4], [5]), but without focusing on the presence of anomalous data, and do not deal with the presence of missing values in the labels.

In this paper, we focus on how anomalies on the labels introduced during measurements in the dataset creation phase can affect ML regression models in the context of MCU performance screening (but perfectly extendable to all the types of regression problem in which we have missing or noisy values in the labels). Once that anomalous labels are recognized and removed from the dataset, we will discuss how to recover information from points that present missing values and lack of labels instead of simply dropping them. We will exploit the relationship between the available labels to reconstruct information that are not present in the dataset with the goal of building a bigger training set, which will be used to train more robust machine learning models. This will be done by using imputation techniques on the labels.

The rest of the paper is organized as follows. Section II presents related works on the topic. Section III describes

the characterization process used to derive the dataset for ML. Section IV introduces the concepts of outlier detection and imputation which are the mechanisms for dealing with anomalous data, while in Section V we detail how to use them to optimize the dataset. The experimental evaluation is presented in Section VI. Finally, Section VII draws our conclusions.

II. Related Work

Different approaches to predict the device performance were suggested in recent years [6]–[9]. In [6], the current was used as a very rough estimator of the performances. The use of additional monitoring and warning infrastructure for a speed violation (Razor FFs) was studied in [7] for tracking the signal delays in particular paths at the system clock speed. It allows very accurate monitoring of a specific path, at the cost of a massive overhead of warning FFs on large chips. The use of ML models to correlate structural and functional $F_{\max}$ was first presented in [1]. The use of indirect measures to predict a circuit specification is called 'alternate test' in literature, and has been widely studied for analog circuits [10]–[13]. More precisely, the idea is to learn the mapping between indirect measurements and circuit specifications, and to use only the indirect low-cost measurements to predict device specifications during production testing. Previous works on MCU performance screening were aimed at deriving a model for $F_{\max}$ prediction [4], [5]. In [4], the values of 27 speed monitors coming from wafer sort were correlated to functional $F_{\max}$ measured on more than 4,000 packaged devices extracted from 26 corner-lot wafers; training devices were randomly selected among the available ones. A first step towards the reduction of the training set size was conducted in [5], where three Active Learning (AL) metrics were deployed to build a training set that considers possible process variations. Even though a reduction of the training set size can be achieved by means of appropriate AL techniques, the presence of outliers negatively influences the overall training procedure. Moreover, the work in [5] does not deal with possible missing or dirty values in the labels.

Working with anomalous data is a well-known topic in the ML community. Anomalous data is expressed by the concept of outlier: "an outlier is an observation which deviates so much from other observations as to arouse suspicions that it was generated by a different mechanism" [14]. Various outlier detection approaches from a data mining perspective are compared in [15], and in particular the classic outlier approaches are resumed, which include statistical-based approach [16], [17], distance-based approach [18], [19], deviation-based approach [20], and density-based approach [21]. We make use of statistical methods to identify outliers both during the label acquisition phase and during the dataset preparation for ML training process.

The definitions and sources of label noise in the context of supervised classification tasks are presented in [22]. The paper proposes a taxonomy of the types of label noise and discusses the potential consequences. It proposes filter approaches to improve the quality of the training data. In such a case, noisy labels are typically identified and being dealt with before training. Mislabeled instances can either be relabeled or simply removed. However, this work deals with classification, rather than regression. It also refers to single-label tasks rather than multi-label and does not deal with missing labels (i.e., samples are simply removed). In this context, we will make use of the correlation among the labels to try to impute missing values and recover information from incomplete data, while we apply anomaly detection techniques directly on the labels.

III. Characterization process

The characterization process for MCU performance screening requires collecting a suitable dataset for the ML training process. The SMONs' data are the features in the ML context, and are measured during the production with high accuracy. Such a measurement is a stable and fast process; hence, the gained features have a high quality. Conversely, the labeling process is a time-consuming procedure performed mostly manually, in which each MCU is measured individually with functional test patterns. The MCU starts to execute the functional pattern with a low frequency, then the frequency is slowly stepped-up in each loop until the functional pattern fails [23]; the resulting threshold frequency $F_{\max}$ is stored. The procedure is typically repeated using various functional test patterns, thus leading to a multi-label dataset. For each MCU, the $F_{\max}$ values collected with the various patterns can have a considerable spread. The most critical pattern is the one with the lowest $F_{\max}$ value, and this is not necessarily the same for all MCUs.

The accuracy of the models and the performance prediction quality are strongly dependent on the quality of the features and labels. While, as mentioned above, the features are acquired with high quality, the labeling might be affected by uncertainty. The measurement of the label is performed under the worst-case voltage and temperature conditions allowed in the datasheet. Minor changes in such measurement conditions have an high impact. The MCU is mechanically mounted on a measurement board that mimics the customer application. Also, here some uncertainty is introduced, such as mechanical vibrations, contact resistance (MCU to board), and white noise. Besides such uncertainty, the MCU itself also widely varies. The main reasons are: (1) the process variation in the CMOS manufacturing and (2) devices with a defect. Every CMOS manufacturing has a particular process variation which is usually follows a Gaussian distribution [24]. This means a vast amount of devices are within the expected process variation. However, some devices are in the tail region of the distribution. Those devices can be determined as outliers. The devices with a defect can also be classified as outliers. There are test methods in the production flow (e.g., based on Stuck-At, Transition, or Path-Delay testing) trying to identify such defective devices. However, some MCUs with certain defects might escape the test procedure, or the tests are still in the engineering phase and not mature enough.

The boxplot in Fig. 1 shows how $F_{\max}$ varies for the set of devices from a wafer. One device is clearly below the threshold value highlighted in the figure for several functional

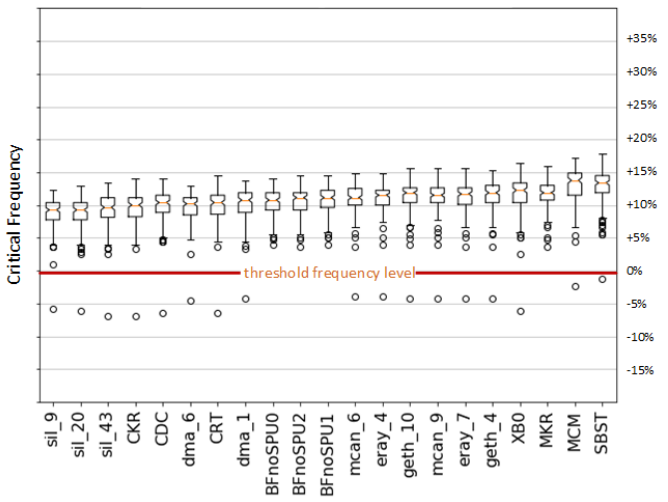

Fig. 1. An overview of functional patterns and their critical fail frequency with the threshold frequency level of the data-sheet. Some outliers are visible over all applied patterns [4]. The threshold is 0%.

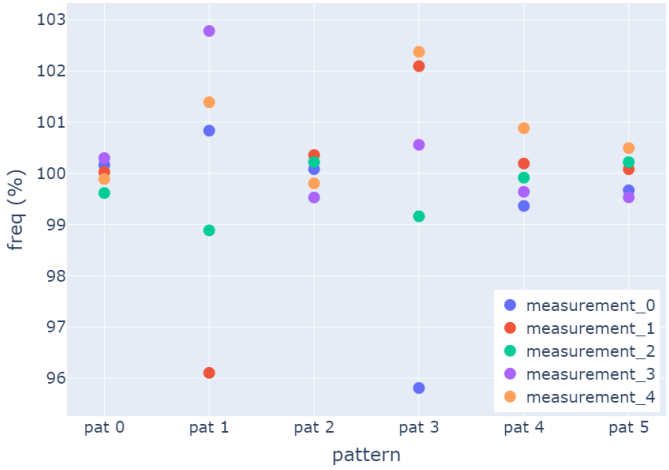

Fig. 2. Graphical view of the five measurements for each patterns of a device

test patterns; other devices are just below the whiskers. Those devices that deviate from the wafer median by several standard deviations in a functional test pattern can be classified as *outliers*. In some cases, a device is classified as outlier for each functional test pattern, while in some others it would be an outlier only for one or few functional test patterns (i.e., the remaining labels are within the expected distributions). An outlier detection procedure is generally applied to each label. Then, in the baseline approach, the device is removed from the training set if an outlier is detected in at least one functional test pattern. This outlier detection is a very pessimistic approach, but necessary for high-quality training data. If only one test pattern is faulty, it does not necessarily mean that the whole device is faulty; nonetheless, all captured labels for that device are discarded. Thus, some mechanisms in the measurement process are integrated to make the label acquisition more robust. Each determination of the $F_{\max}$ is repeated multiple times (five in our case study) for every device; the median value is calculated and determined as the final $F_{\max}$ value. If a device has a high variance between the various measurements, the label is treated as an anomaly and discarded, introducing a *missing value* in the dataset. In Fig. 2, we can see a graphical view of the repeated measurements of a device as an example. The labels for the patterns number 1 and number 3 would be discarded due to the high variance among the measurements (on the y-axis, the deviation of the measurement w.r.t. the mean for that pattern).

The dataset after the (imperfect) labeling process contains some outliers (devices anomalous with respect to the wafer median) and some missing values (patterns which repeated measurements present high variance). Simply discarding all dubious devices which might have an uncertainty will significantly reduce the training set. Methods are needed to cope with such a dataset.

IV. Dealing with Anomalous Data

For a ML algorithm, data are like gold, and the more data we have the more likely will be our algorithm to reach a high generalization capability. In the context of MCU performance screening, the low availability of labelled data and the high cost in obtain them lead us to exploit all possible information retrievable from the data. Even an incomplete sample (that misses some values) may carry useful information, if handled with care. But data must also be of high quality. A ML model trained with a properly chosen dataset, clean by noise and outliers, can achieve the same performances (if not better) of a model trained on a larger amount of data but of poor quality. In the following, we will introduce the statistical methods used for optimizing the training set, thus reducing the amount of data required to reach a sufficient accuracy in the final ML models.

A. Outlier Detection

Outlier detection is a critical task in many data-science fields. Outliers indicates abnormal conditions that may cause significant performance degradation in ML models. Example of outliers can be: abnormal objects in images, intruder inside a system with malicious intentions but also faults on production lines. The effect of outliers and noise in data measurements phase involves a shift in the underlying data distribution: if data are affected by noise, the distribution of inputs $p(x)$, of labels $p(y)$, and the joint distribution $p(x, y)$ of labels and inputs may vary. As a consequence, an ML model might learn something different with respect to information that can be retrieved from data, and in particular a different discrimination distribution $p(y|x)$ with respect to the one related to the underlying processes.

Outlier detection has been classified in three types [25]. In Type 1, we perform outlier detection with no prior knowledge of the data; This type is based on a clustering approach. In Type 2, the aim is to model both normality and abnormality; This approach is analogous to supervised classification and requires pre-labelled data for the normal and abnormal class. In Type 3, the aim is to model only normality; the normal

class is given and an algorithm will autonomously learn to recognise abnormality by defining boundaries of normality; authors generally name this technique novelty detection or novelty [25]. Our approach is a mixed Type 1 and Type 2 outlier detection, since we make use of statistical tools to detect abnormal data, but we have some pre-existing indication on the outlierness of samples. Statistical approaches are based on metrics like interquartile range (IQR) or z-score.

IQR is an outlier detection technique based on considering as outliers each sample outside a specific range, computed on the basis of first and third quartile (Q_1 and Q_3) of the data distribution. This method goes trough these steps: (1) calculate Q_1 and Q_3, and (2) compute the IQR as $Q_3 - Q_1$ and the lower and upper bound (lb and ub, respectively) as:

$$lb = Q_1 - 1.5 \cdot IQR \qquad ub = Q_3 + 1.5 \cdot IQR$$

Anything outside the range $[lb, ub]$ is considered an outlier [26]. The factor 1.5 may vary in some implementations.

The z-score is a numerical measurement that describes a the relationship between a sample and the mean of a group of samples. A z-score is measured in terms of standard deviations from the mean. The formula for the z-score of a sample is $z = \frac{x-\mu}{\sigma}$, in which μ and σ are the mean and the standard deviation of the population. A z-score equal to 3 means that a sample lies 3 standard deviations above the mean. If the samples are normally (i.e., Gaussian) distributed, it is possible to compute the confidence interval (and relative standard normal table). The probability that a sample has a z-score below 3 is approximately 99.865%. This means that a sample has a probability of just 0.135% to finish above 3 standard deviations. Thus, 3 will be used as the z-score limit value to discriminate among inliers and outliers in our experiments.

B. Imputation

Missing values may be present both in the feature space and in the label space. If a label is missing in a single-label dataset, the corresponding sample cannot be used in supervised algorithms, and thus it is typically discarded. However, when dealing with a dataset containing missing values on features (or, in multi-label datasets), removing samples which may be valuable for successive learning algorithm is not always acceptable, especially when the effort for collecting those samples is high. A better strategy is to try to recover missing values by resorting to mathematical procedures like weighting or imputation [27], which try to infer missing values from the known part of the data. One type of imputation algorithm is the Univariate Imputation, which imputes the missing values present in a column of the dataset by using only non-missing values in that feature dimension (for example, by replacing the missing value with the mean value of that feature, or with the median or another standard properly-chosen value). Multivariate imputation algorithms [28], instead, use the entire set of available features to estimate the missing values. A missing value in one column of the dataset is imputed on the basis of all the other columns. An example of Multivariate Imputation is the Iterative Imputer.It iterates for each feature and, at each step, a feature column is designated as output

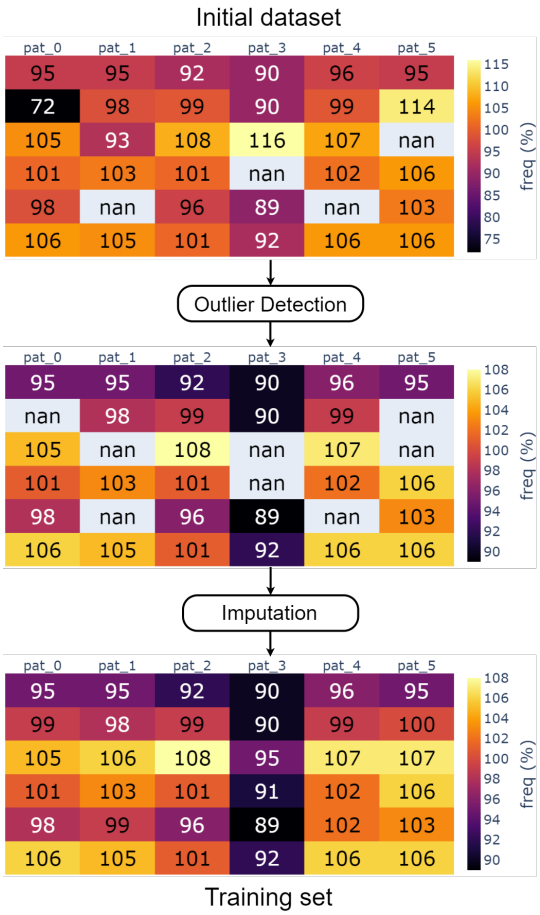

Fig. 3. Step performed during the data preparation phase on the labels dataset.

y and the other feature columns are treated as inputs X. A base regressor is fit on (X, y). Then, the regressor is used to predict the missing values of y. This is done for each feature in a round-robin fashion. The imputation procedure is repeated for a certain number of iteration. The results of the final imputation round are returned.

V. Optimizing the Training Set

During the characterization phase, multi-label samples are collected and added to the dataset. We aim at identifying anomalous and noisy data from the part of the dataset used for the training process. As a result, the optimized dataset lets us stop the characterization earlier. Hereinafter, we will refer to such a training set as the dataset intereset by the optimization. The procedure consists of three steps, which will be explained using a simplified dataset and graphically visualized in Fig. 3, without features. For each of the 6 samples in the toy dataset, 6 distinct labels are collected using multiple measurements per label. Frequency values in the figure are normalized on the mean of the corresponding label. In the example, we will focus on labels, but the approach can be extended to features.

In the first step (upper dataset in Fig. 3), noisy data are identified by looking at the variance of the multiple measurements of each label (see Fig. 2). Labels with a high variance are dropped and replaced with a not-a-number (*NaN*), thus

introducing a missing value in the dataset. This first step is a "filter" applied by domain experts, and the threshold value depends on the quality of the measurement process. Outliers can escape from this procedure, justifying the use of additional detection methods.

The second step consists of applying statistical methods for outlier detection, based on IQR and z-score (see Section IV-A), to discover labels measurements that are out of the normal distributions (see Fig. 1). The analysis is performed per label, and can be eventually performed per wafer (i.e., in that case, when computing IQR or z-score, only the samples in the same wafer would be considered). In the upper dataset of Fig. 3, the second sample shows two outlier labels (the first and the last pattern), which are replaced with *NaN* values in the middle dataset. If most of the labels of a given sample are *NaN*, then that sample should be discarded.

In the last step, we use imputation to fill the missing values. This permits to avoid to drop samples and to recover information also by incomplete rows. But simple univariate imputation, that replaces the missing values of a feature with the mean or median of that feature in the dataset, could not be the correct choice, resulting into a too simplistic approach that may introduce further noise in the data. Also, when labels are highly correlated, as in the problem at hand, it is absolutely reasonable to build a model, even a simple linear one, able to predict the value of a target on the basis of the others. This justifies the deployment of a multivariate imputation model, able to predict missing values relying on multiple columns of the dataset. The resulting values are visible in the lower dataset of Fig. 3. Here we are considering both missing and removed data in the same way, due to the fact that (as explained in Section III) devices which repeated measurements present high variance are discarded from the dataset in advance.

VI. Experimental Evaluation

A. Experimental Setup

The proposed methodology has been validated on a dataset composed of 4,039 devices, with 27 features (SMONs) and 10 labels (functional test patterns), plus an extra label taking the minimum among the patterns, for each device. This is the final target in the performance screening procedure (the critical pattern of each device). The test set used for the evaluation of all models includes 602 devices without any missing label, randomly sampled but stratified per-wafer. Such a test set is the same used in [4], [5], as a comparison. Alternatively, cross-validation can be used to better estimate the generalization error on several test sets. The training set of each model is derived by applying the proposed methodology on the remaining 3,437 devices, which include 2,058 full-row samples (i.e., they do not present missing values) and 1,632 samples with at least one missing value due to the high variance in the repeated F_{max} measurements. In the baseline approach [4], anomalous data were identified in 423 devices by the domain experts who acquired the labels, and those samples were marked as outliers. Outliers were discarded from the training set used in the baseline approach while they were kept in the test set; the resulting baseline consists of only 1,805 full-row samples – less than 53% of the available devices. For the imputation procedure, we chose the Iterative Imputer of scikit-learn [29] with Random Forest as the base estimator to mimic the MissForest imputation algorithm [30], but also a simple linear regression would fit well in our case, with comparable performances. The regression algorithms used are 1) the Ridge Regression with a a pre-processing pipeline of normalization of input, PCA feature reduction with 14 components, and polynomial transformation and 2) the Kernel Ridge Regression with input normalization and the radial basis function (RBF) kernel. Both the algorithms work in a multi-output fashion using a Regression Chain approach [31].

We chose to combine z-score and IQR, in a double-level check fashion, marking as *NaN* the labels that present a z-score higher than 3 or that are outside the IQR range. These procedures can be done at the whole-dataset level or per-wafer. This last wafer-level approach considers the process variations of the corner-lots and the presence of faster and slower wafers. In our experiments, we obtained the best results using this last approach.

B. Results

In Table I, we resumed the performances of the same ML algorithm trained on various training set, derived by using different outlier detection and imputation schemes. The first three columns of the table describe how the training set has been derived. Column 1 (*Missing values*) indicates whether we have utilized *imputation* on the missing values or we simply *drop* the corresponding sample. Similarly, column 2 (*Baseline outlier*) tells whether, in presence of anomalous data identified by the baseline approach [4], to *keep* the original values, to replace them with *NaN* and apply *imputation*, or to *drop* the sample. Column 3 (*Outlier detection*) indicates whether additional outliers on labels are identified at *dataset-level* or at *wafer-level* (*None* indicates that this step is not performed). Column 4 (*Training set size*) shows the number of samples used for the training. Finally, the last two columns report the normalized root mean squared error (*nRMSE*) and mean absolute error (*nMAE*) of the model on the test set.

The first two models in Table I do not make use of the proposed approach. The second case represents the baseline model, which shows how removing the baseline outliers helps to increase the model performance. The three models in the following group of lines show imputation combined with outlier detection; in that case, 100% of the available samples can be used for the training. The following two groups of lines also highlight that baseline outliers should be subject of imputation or discarded to outperform the baseline model; in the last case, we use around 88% of the available samples. In the final group of models, we show that a mixed solution, where we drop the devices than present a number of missing labels greater than a certain *threshold* (5 in our case) and impute the others, we can further increase the performances; these last models use 87% of the available samples or more. This final approach is based on the idea that the imputation procedure is considered unreliable when the number of available labels is low, even though the correlation among them is practically perfect. Similar results can be obtained both for

TABLE I
PERFORMANCES ON TEST SET WITH DIFFERENT METHOD FOR PATTERNS OUTLIER DETECTION (POLYNOMIAL RIDGE REGRESSION)

	Missing values	Baseline outliers	Outlier detection	Training set size	nRMSE	nMAE
0	Drop	Keep	None	2,058	1.67%	1.27%
1*	*Drop*	*Drop*	*None*	*1,805*	*1.59%*	*1.19%*
2	Imputation	Keep	None	3,437	1.66%	1.28%
3	Imputation	Keep	Dataset-level	3,437	1.66%	1.28%
4	Imputation	Keep	Wafer-level	3,437	1.58%	1.20%
5	Imputation	Imputation	None	3,437	1.58%	1.20%
6	Imputation	Imputation	Dataset-level	3,437	1.57%	1.19%
7	Imputation	Imputation	Wafer-level	3,437	1.59%	1.21%
8	Imputation	Drop	None	3,014	1.57%	1.18%
9	Imputation	Drop	Dataset-level	3,014	1.56%	1.16%
10	Imputation	Drop	Wafer-level	3,014	1.57%	1.18%
11	Threshold	Imputation	Wafer-level	3,343	1.60%	1.22%
12	Threshold	Drop	Wafer-level	2,986	1.56%	1.17%

* Baseline model [4]

linear and kernel-based algorithms (with nRMSE of 1.56% and 1.50%, respectively). The Coefficient of determination R2 is equal to 0.9842 for the baseline (1) and 0.9846 for the best approach (12). The increased number of samples permits to have more robust models in the sense of portion of input-space explored. We can notice that models 5-7 and 8-10 are similar: with imputation, no matter how the baseline outliers are corrected (drop/imputed), the nRMSE and nMAE are very close to each other. In this case, the imputation seems not to add relevant information. Since the values of these labels are in the distribution's tails, the information created by the imputation on the basis of the remaining pattern may be irrelevant. Keeping those samples in the training set may alter the training procedure. We compared the proposed most promising strategies and the baseline model using the learning curves for both regression algorithms, as shown in Fig. 4. The x-axis represents the number of devices characterized and available in the best case (i.e., when 100% of them are used for the training); the percentage of actual samples in the training set with respect to the available devices depends on the strategy used and reflects the results in Table I. The y-axis reports the nRMSE of the model on the test set. All models were evaluated on the same test set, at each step of the learning curve creation. The learning curves show that we can reduce the number of devices that need to be characterized to reach a certain amount of accuracy. Let us focus on Fig. 4: for the Ridge regression (upper figure), the yellow curve (corresponding to the model 12 in Table I) reaches the final accuracy of the baseline method (i.e., 1.59% of nRMSE, red curve) just after 1,000 samples, and remains stably below it. This means that just 1,000 samples might have been characterized for the training, that is less than 30% of the devices needed for the baseline approach. Similar trends can be observed with the Kernel Ridge regression (lower figure). Hence, we can save a great amount of human-time needed for acquiring the measurements of the labels.

To resume, the imputation technique have the double advantages of 1) minimizing the number of samples dropped, increasing both the training set size and the performances and building robust algorithm and 2) reducing the number of devices to be characterized to reach satisfactory final accuracy, taking advantage of the different labels available and reducing the time measurement cost.

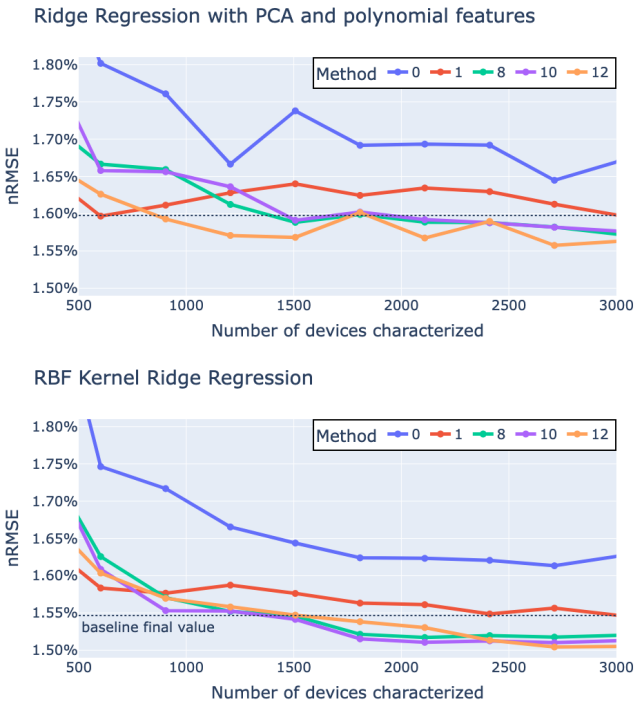

Fig. 4. Learning curves of Ridge (upper plot) and Kernel Ridge (lower plot) models with different types of training sets, tested on the same test set. On the x-axis, the number of devices characterized (constant), on the y-axis the nRMSE. The dashed lines are the nRMSE values reached by the baseline models (1, red) with the maximum number of samples in the training sets.

VII. CONCLUSIONS

We presented techniques for dealing with anomalous and noisy values in the labels in the context of MCU performance screening. The detection of outliers is a fundamental step as they affect the performances of ML models. Results showed that imputation is an appropriate approach in this context, since it can be helpful to increase the number of available samples for the training procedure. We were able to increase the quality of existing methods for MCU performance screening, with the double goal of 1) creating a dataset with a higher quality, reaching 1.56% of nRMSE using a linear regression algorithm and 2) reducing the number of devices to be characterized by a factor of three with respect to the baseline approach. This work can be extended to other contexts that present errors in measurements, noise, and high-correlated labels.

REFERENCES

[1] J. Chen *et al.*, "Data learning techniques and methodology for Fmax prediction," in *2009 ITC*, 2009.
[2] J. Chen *et al.*, "Selecting the most relevant structural Fmax for system Fmax correlation," in *2010 VTS*, 2010.
[3] M. Sadi *et al.*, "SoC Speed Binning Using Machine Learning and On-Chip Slack Sensors," *IEEE TCAD*, 2017.

[4] R. Cantoro *et al.*, "Machine Learning based Performance Prediction of Microcontrollers using Speed Monitors," in *2020 ITC*, 2020.

[5] N. Bellarmino *et al.*, "Exploiting active learning for microcontroller performance prediction," in *2021 IEEE European Test Symposium (ETS)*, 2021.

[6] K. von Arnim *et al.*, "An effective switching current methodology to predict the performance of complex digital circuits," in *2007 IEEE International Electron Devices Meeting*, 2007.

[7] D. Blaauw *et al.*, "Razor II: In situ error detection and correction for PVT and SER tolerance," in *2008 IEEE International Solid-State Circuits Conference - Digest of Technical Papers*, 2008.

[8] G. Sannena *et al.*, "Low overhead warning flip-flop based on charge sharing for timing slack monitoring," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 2018.

[9] T. B. Chan *et al.*, "DDRO: A novel performance monitoring methodology based on design-dependent ring oscillators," May 2012.

[10] H. Ayari *et al.*, "Making predictive analog/rf alternate test strategy independent of training set size," in *2012 IEEE International Test Conference*, 2012.

[11] P. Variyam *et al.*, "Prediction of analog performance parameters using fast transient testing," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2002.

[12] H.-G. Stratigopoulos *et al.*, "Error moderation in low-cost machine-learning-based analog/rf testing," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2008.

[13] J. Brockman *et al.*, "Predictive subset testing: Optimizing ic parametric performance testing for quality, cost, and yield," *IEEE Transactions on Semiconductor Manufacturing*, 1989.

[14] D. M. Hawkins, *Identification of outliers*, English. Chapman and Hall London ; New York, 1980.

[15] J. Xi, "Outlier Detection Algorithms in Data Mining," in *2008 Second International Symposium on Intelligent Information Technology Application*, 2008.

[16] A. Barnett *et al.*, "On the estimation of common non-linearity among repeated time series," in *Proceedings of the 11th IEEE Signal Processing Workshop on Statistical Signal Processing (Cat. No.01TH8563)*, 2001.

[17] P. J. Rousseeuw *et al.*, *Robust Regression and Outlier Detection*, ser. Wiley Series in Probability and Statistics. Wiley, 1987.

[18] E. M. Knorr *et al.*, "Distance-based outliers: Algorithms and applications," *The VLDB Journal The International Journal on Very Large Data Bases*, 2000.

[19] S. Ramaswamy *et al.*, "Efficient Algorithms for Mining Outliers from Large Data Sets," in *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '00, Dallas, Texas, USA: Association for Computing Machinery, 2000.

[20] A. Arning *et al.*, "A Linear Method for Deviation Detection in Large Databases," in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, ser. KDD'96, Portland, Oregon: AAAI Press, 1996.

[21] M. M. Breunig *et al.*, "LOF: Identifying Density-Based Local Outliers," in *ACM SIGMOD*, Dallas, Texas, USA, 2000.

[22] B. Frenay *et al.*, "Classification in the Presence of Label Noise: A Survey," *IEEE Transactions on Neural Networks and Learning Systems*, 2014.

[23] R. McLaughlin *et al.*, "Automated Debug of Speed Path Failures Using Functional Tests," in *2009 27th IEEE VLSI Test Symposium*, 2009.

[24] K. Maragos *et al.*, "In-the-Field Mitigation of Process Variability for Improved FPGA Performance," *IEEE Transactions on Computers*, 2019.

[25] V. Hodge *et al.*, "A survey of outlier detection methodologies," *Artificial Intelligence Review*, Oct. 2004.

[26] L.Sunitha *et al.*, "Automatic Outlier Identification in Data Mining Using IQR in Real-Time Data," *International Journal of Advanced Research in Computer and Communication Engineering*, 2014.

[27] J. Brick *et al.*, "Handling missing data in survey research," *Statistical Methods in Medical Research*, 1996, PMID: 8931194.

[28] S. van Buuren *et al.*, "mice: Multivariate Imputation by Chained Equations in R," *Journal of Statistical Software*, 2011.

[29] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *Journal of machine learning research*, 2011.

[30] D. J. Stekhoven *et al.*, "MissForest—non-parametric missing value imputation for mixed-type data," *Bioinformatics*, Oct. 2011.

[31] E. Spyromitros-Xioufis *et al.*, "Multi-label classification methods for multi-target regression," *arXiv*, Nov. 2012.