Impact of Feature Selection and CIR Window Length on NLoS Classification for UWB Systems

Elisei Ember*†, Jesus Pestana*, Michael Krisper*†, Michael Stocker†, Kay Römer†,
Carlo Alberto Boano†, and Pablo Corbalán Pelegrín‡

*Pro2Future GmbH, Graz, Austria †Institute of Technical Informatics, Graz University of Technology, Graz, Austria
‡NXP Semiconductors Austria GmbH & Co KG, Gratkorn, Austria

Email: {name.surname}@pro2future.at ; {michael.stocker, roemer, cboano}@tugraz.at ; pablo.corbalanpelegrin@nxp.com

***Abstract*—Indoor localization systems based on Ultra-WideBand (UWB) technology can typically achieve cm-level accuracy, but their performance degrades in Non-Line-of-Sight (NLoS) conditions. To cope with this problem, Machine Learning (ML) techniques have been applied to detect such NLoS conditions and adapt the localization algorithm accordingly. However, such ML techniques are typically optimized for accuracy, resulting in computationally-complex models that cannot be run on resource-constrained UWB devices. In this paper, we study and propose methods to reduce the computational complexity of NLoS classification models by applying ML-based feature selection and by reducing the window length of the channel impulse response for feature extraction. Specifically, we consider 29 features and study the effect of feature selection across five different datasets to obtain generalizable results. We show that we can extract two sets of only 3 and 8 features, which result in tiny ML models (smaller than 1 kB), and low computation times (3.6 ms and 27.7 ms on a 80 MHz ESP8266 microcontroller, respectively). This allows a reduction of the runtime by more than 90% compared to the state of the art, while still maintaining an average classification accuracy above 85% across all five datasets.**

***Index Terms*—Channel impulse response, NLoS classification, Machine learning, Feature selection, Embedded systems, Ranging.**

I. Introduction

The popularity of UWB systems is soaring, thanks to their increasing adoption in industrial applications [1], as well as the introduction of UWB radios into modern smartphones [2] and vehicles [3]. UWB systems enable cm-accurate localization in indoor settings by means of multi-node ranging and triangulation. In order to achieve such high-accuracy localization, UWB systems employ a large bandwidth (≥ 500 MHz) and transmit short signal pulses (≈ 2 ns). This allows UWB receivers to accurately estimate the time-of-arrival of a signal and, consequently, the distance between two UWB devices performing peer-to-peer ranging, by inferring the so-called Direct Path (DP) from the estimated Channel Impulse Response (CIR). In simple terms, for Line-of-Sight (LoS) conditions, the DP is identifiable by the First Peak (FP) in the CIR, which represents the first path that arrived to the receiver.

The focus of our work is the classification of LoS and Non-Line-of-Sight (NLoS) conditions during peer-to-peer UWB communication. Under normal LoS conditions, i.e., when no obstacles block the DP between two devices, the FP is clearly distiguishable from other Multi-Path Components (MPCs) originating from reflections by walls and other objects (Fig. 1, top). However, in NLoS conditions, the direct path between transmitter and receiver may be blocked or attenuated, resulting in a decreased amplitude of the FP compared to the MPCs, or it can even drop under the noise threshold (Fig. 1, bottom). As a result, an MPC may be erroneously identified as FP, causing ranging errors up to a few meters [4], [5].

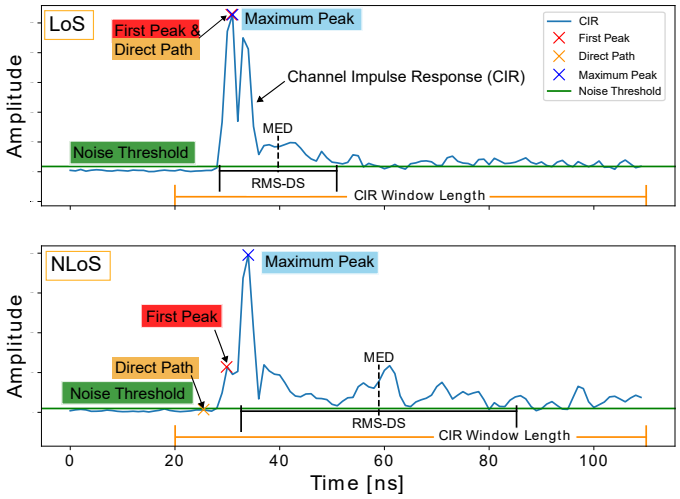

Fig. 1. Channel Impulse Response (CIR) estimated by an UWB radio in a Line of Sight (LoS) condition (top) and a Non-Line of Sight (NLoS) condition (bottom), where an obstacle blocks the Direct Path (DP) of the UWB signal. Some characteristics are highlighted, e.g., the CIR Window Length (CIR-WL) and the noise threshold, which are used to extract the features for the NLoS/LoS scenario classification (see Table I, §II). The CIR-WL used for extracting features, here 90 ns, starts 10 ns before the estimated DP, which is the First Peak (FP) in the CIR. Often, to take into account the undersampling of the CIR, the position and amplitude of the FP are further refined, e.g., by means of interpolation. In the shown NLoS case, the peak corresponding to the DP is completely attenuated. In both plots, the features of Mean Excess Delay (MED), RMS Delay Spread (RMS-DS), First Peak (FP) and Max. Peak (MP) are shown. The plots showcase common differences on the feature values for LoS and NLoS cases. In LoS cases, the MED is commonly close to the FP, and the RMS-DS is usually narrower than in NLoS cases. The configuration and the device determine the units of measurement of the CIR signal amplitude.

NLoS detection. To tackle this problem, the research community has developed many solutions to differentiate between NLoS and LoS scenarios [6], [7]. This information can be used to mitigate ranging errors [5], or to improve the localization accuracy by means of a refined anchor selection [8], [9]. Several methods leverage the CIR for NLoS

classification, and either follow statistical approaches [10], [11] or make use of Machine Learning (ML). A notable ML-based approach is the one proposed by Maranó et al. [4], who extract six features[1] from the CIR and use them to train a Least-Squares Support-Vector Machine (LS-SVM) to classify NLoS and mitigate ranging errors. Building upon this seminal work, Stocker et al. [5] introduce additional features, benchmark their performance for NLoS classification, and apply their results to improve the correction of ranging errors. Nguyen et al. [13] compare LS-SVMs against Relevant Vector Machines (RVMs), showing better performance. Ferreira et al. [14] use 14 features provided by the popular Qorvo DW1000 radio [15], perform ML feature selection using correlation matrix-based analysis, and utilize several ML models for NLoS classification as well as regression algorithms for ranging error mitigation. Recently, there has been an increasing interest in Deep Neural Networks (DNNs) [16], [17], as these do not rely on manually-crafted features, but instead learn features autonomously during training.

Limitations of existing solutions. Existing works on NLoS classification mainly focus on achieving high accuracy and rely on computationally-expensive ML classifiers or a high number of features [5], [12], [17], which do not fit the constraints of the low-power UWB tags used to build location-aware IoT applications (e.g., the off-the-shelf Qorvo `MDEK1001` device runs at 64 MHz and embeds only 256/32 kB of flash/RAM). Existing solutions are typically meant to run on powerful edge devices [17]–[19], e.g., Raspberry Pis featuring fast CPUs/GPUs and Gigabytes of memory. This, however, requires the transmission of the CIR to an edge device and of the classification result back to the UWB tag, which increases the requirements on network bandwidth and results in poor scalability and reliability (e.g., radio interference from nearby Wi-Fi devices can destroy or corrupt the transmitted packets [20]).

For these reasons, NLoS classification should ideally be performed directly on the UWB tag itself [9], [21]. This, however, requires ML models that are not only *tiny* (i.e., with a small memory footprint), but also *lightweight* (i.e., computationally simple). In fact, a short runtime of the ML model is essential to be able to update the tag's position at a high frequency. Unfortunately, the state-of-the-art (SoA) has only devoted limited attention to the creation of small and lightweight models [9]. In particular, we identify an important gap to fill: none of the existing studies in the literature reports the feature extraction times, which is crucial information when designing localization solutions sustaining high update rates on embedded micro-controllers. Moreover, existing literature typically relies on algorithms learning the relevant features for classification automatically (e.g., DNNs [16]–[18]) or on a relatively small numbers of features [14], [22], [23], which does not shed light on a small set of features enabling a high classification accuracy. Existing works also typically focus on only one dataset, which prevents a *generalization* of findings. In fact, as we show in this paper – in which we systematically benchmark a large number of ML classifiers over 5 different datasets – the performance of a feature may vastly vary across datasets due to the presence/absence of specific corner cases.

Contributions. We hence focus on NLoS classification for resource-constrained UWB devices, aiming to derive tiny ML models and techniques to reduce their computational complexity (e.g., feature selection and shorter feature extraction time).

We start from a set of 29 features (29 Fs), which is – to the best of our knowledge – larger than the most comprehensive study in the literature [12], and identify a suitable method to reduce the number of features while delivering a high classification accuracy. Feature selection[2] is a broad topic in ML [24]. In our investigation, we have found particularly interesting the application of Decision Trees (DTs) in other fields of research [25], [26], due to their capacity to deal with higher numbers of features and relatively lower training times. We perform our feature selection procedure (described in §II), and compare the resulting feature importance rankings with the SoA in §III-A. Based on the feature rankings, we propose small sets of only 3 and 8 features to build NLoS classification models that consistently achieve a high accuracy across several datasets. This small sets of features yield dramatic reductions of the feature extraction time (FE-Time): from 90 ms (for 29 Fs) to 45.8 ms (for 8 Fs) and 6.21 ms (for 3 Fs).

We further analyze how to reduce the FE-Time by shortening the CIR Window Length (CIR-WL) considered for feature extraction, an aspect that was investigated by related work only to a limited extent [9], [18], [27]. Specifically, we first design an ML Pipeline (described in §II) to automatically select features with a single CIR-WL, from a range of possible CIR-WL values, by means of hyperparameter tuning. Additionally, we propose a method to select features sets with features extracted using different CIR-WLs. To this end, we add a step to our feature selection process and select the best CIR-WLs using the Kolmogorov-Smirnov statistic (showcased in §III-B), a method already applied in other fields of research [28]. This way, we achieve even faster FE-Times of 27.7 ms (for 8 Fs) and 3.61 ms (for 3 Fs) – which represents a reduction by more than 90% compared to the SoA – while maintaining a classification accuracy (between 76.6% and 91.5%) that is in line with or even superior than the SoA, as shown in §III-C.

Therefore, our findings enable the creation of NLOS classification schemes that are accurate, tiny, and offer fast FE-Times, which are a good fit for resource-constrained Micro-Controller Units (MCUs) such as the 80 MHz `ESP8266`. Note that our findings are evaluated across four publicly-available datasets and our own INDUSTRIAL dataset, which shows the generality of the proposed approach and provides a basis for comparison with previous SoA methods.

[1]Examples of features include the calculations of statistics from sequences of Received Signal Strength (RSS) samples [12] or the calculation of certain characteristics of the CIR estimate [4], [5], [12] such as the Signal-to-Noise Ratio (SNR), i.e., the ratio of the signal power to the noise power.

[2]Several feature selection methods have been used for NLoS classification, such as *(i)* adding features incrementally in different combinations [4], [5], *(ii)* using correlation matrices estimating the amount of information provided by the features [14], and *(iii)* applying genetic algorithms [12].

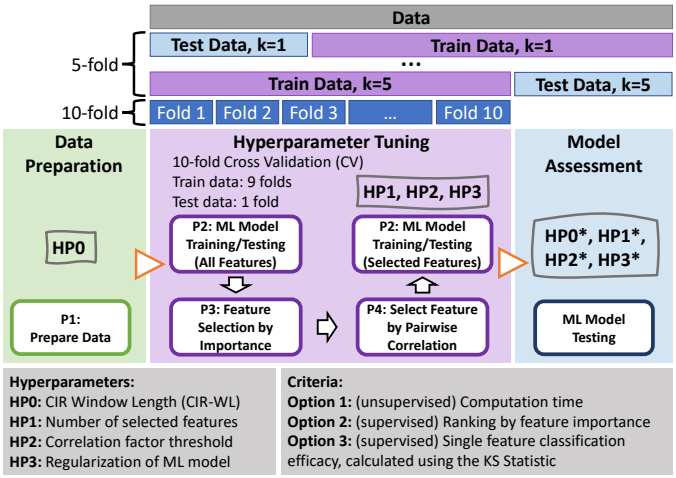

Fig. 2. ML pipeline proposed in this work, which incorporates best ML practices, hyperparameter tuning, and assessment via nested cross-validation.

II. Methodology

We propose an automated ML pipeline for the training of ML models for the classification of UWB NLoS/LoS conditions. Our method follows two main approaches to reduce the computational complexity: (1) feature selection, where we identify and use only the most relevant features for classification, and (2) reduction of the feature extraction time, achieved by shortening the CIR Window Length (CIR-WL) for feature extraction. For a fair comparison with the SoA, we use a wide set of features, implement our models using several ML classifiers, and evaluate them across 5 datasets.

ML pipeline. Our proposed ML pipeline (Fig. 2) implements Nested Cross-Validation (Nested CV) [29]–[32], so that we can perform hyperparameter (HP) tuning and obtain unbiased performance estimates for our ML models. The HP tuning uses the NLoS/LoS classification accuracy as optimization objective to select the HPs, namely, a single CIR-WL (HP0) for the feature extraction, the feature selection HPs (HP1, HP2), and the ML model regularization parameters (HP3). As shown in Fig. 2, Nested CV allows us to achieve the HP tuning in the inner CV loop ($k_{inner} = 10$ folds). After re-training with the tuned HPs, the outer CV loop assesses the performance of the ML model in a hold-out fold ($k_{outer} = 5$ folds). These k_{outer} assessments provide an unbiased estimate of the performance of the final ML model during deployment [30]–[32].

Data preparation. Our CIR-WLs start 10 ns before the estimated DP, which is the FP in the CIR. The feature extraction and a (per-feature) min-max normalization per each CIR-WL (HP0) are part of the data preparation (Fig. 2).

To tackle the issue of dataset imbalance, we perform stratified splits according to certain parameters provided by the dataset, utilizing the `StratifiedGroupKFold` class from the Python `scikit-learn` library. The group split is performed to obtain train and test sets that are distinct from each other at that level and setting, e.g., room type and frequency channel setting, in order to replicate real-world outcomes. Generally, the *stratified group k-fold split* might yield a lower accuracy compared to a *randomized k-fold split*.

TABLE I
Catalogue of features extracted from the CIR.

Feature Name	Calculation and comments
[F1] RMS Delay Spread (RMS-DS)	$\tau_{RDS} = \left(\sum_0^N (t_k - \tau_{MED})^2 \lvert c(t_k)\rvert^2 / E_{CIR}\, \Delta t_k\right)^{1/2}$ τ_{MED}: Mean Excess Delay (MED). E_{CIR}: Energy. $c(t)$ denotes the CIR signal. $t_{[]}$ or t_k denote timestamps. Δt_k: time interval.
[F2] Signal to Noise Ratio (SNR)	$SNR = \mu P_{signal} / \mu P_{noise}$ μP_{signal} and μP_{noise} are the average power of signal and noise respectively.
[F3, F9, F11 (3 features)] Mean Excess Delay (MED)	[F11] $\tau_{MED} = \sum_0^N t_k \cdot \lvert c(t_k)\rvert^2 / E_{CIR}\, \Delta t_k$ Extra Fs taking t_L (Low threshold) or t_{FP} (First Peak) as initial time: (1) [F3] $\tau_{MED\_L} = \tau_{MED} - t_L$ and (2) [F9] $\tau_{MED\_FP} = \tau_{MED} - t_{FP}$.
[F4] Energy of the CIR Signal	$E_{CIR} = \sum_1^N \lvert c(t_k)\rvert^2\, \Delta t_k$ Its value depends on the CIR window length.
[F5] Form Factor	$k_F = RMS_{CIR} / \mu_{CIR}$ RMS: Root Mean Square value. μ: mean.
[F6] Standard Deviation	σ_{CIR}: (statistics) standard deviation of the whole CIR.
[F7] Rise Time [4]	$\Delta t_{rise} = t_H - t_L$ Elapsed time between 2 threshold values, High (H) and Low (L), of the CIR [4].
[F8] Energy difference total minus FP [5]	$\Delta E_{CIR\_FP} = E_{CIR} - E_{FP}$ $E_{FP} = \sum c(t_k)^2\, \Delta t_k$ for t_k around t_{FP}.
[F10] Skewness	$skew = m_3 / m_2^{3/2} = m_3 / \sigma_{CIR}^3$ Same definitions as in statistics for the kurtosis, skewness and the j^{th} central moments m_j.
[F12, F22 (2 features)] Rician K Factor [33], [34]	$k_{RK} = P_{LoS} / P_{NLoS} = P_{FP} / \sum_1^{N_{MPCs}} P_{MPC\ k}$ [F12] Dominant path to remaining paths power ratio. [F22] Extra feature with P_{MP} (Max. Peak), rather than with P_{FP}. MPCs: Multipath Components.
[F13] Kurtosis	$kurt = \kappa = m_4 / m_2^2 = E\left[(c(t) - \mu_{CIR})^4\right] / \sigma_{CIR}^4$
[F14] First Peak (FP), or first path, amplitude	$C_{FP} = c(t_{FP})$ CIR value at the First Peak (FP) $C_{[]}$ values extracted from CIR.
[F15] FP to total Energy Ratio	$ER_{FP:CIR} = E_{FP} / E_{CIR}$ ER: Energy Ratio
[F16, F19 (2 features)] Mc (by Decawave [23])	$Mc = C_{FPP} / C_{MP}$ With $FPP = \max(FPP_1, FPP_2, FPP_3)$ with FPP_j being the peaks after t_{MP} ordered by occurrence, or around t_{MP} ordered by amplitude.
[F17] FP to MP Energy Ratio	$ER_{FP:MP} = E_{FP} / E_{MP}$
[F18] Maximum Peak (MP), or max. amplitude	$C_{MP} = c(t_{MP}) = \max_t \lvert c(t)\rvert$ MP: Maximum Peak or value.
[F20] Noise Power before the FP	$P_{NBFP} = \frac{1}{N} \sum_{t_L - N}^{t_L} c(t_k)^2\, \Delta t_k$ Summation using only values before t_L. [5]
[F21] Elapsed time between FP and MP [23]	$\Delta t_{MP\_FP} = t_{MP} - t_{FP}$ FP: First Peak or, in some cases, named first path.
[F23, F27 (2 features)] Likelihood of Undetected Early Paths (LUEP) (by Decawave [23])	$LUEP = peak\ count / max\ peaks\ count$ $peak\ count$: number of peaks above noise threshold before t_{MP}, with max being its theoretical maximum value. Extra feature calculated using the range from [F27] t_L or [F23] t_{FP} up to t_{MP}.
[F24] Energy difference between MP and FP	$\Delta E_{MP\_FP} = E_{MP} - E_{FP}$
[F25] Rise time to FP [23]	$t_{rise\_FP} = t_{FP} - t_L$
[F26] Peak to RMS Power Ratio	$PR_{MP:RMS} = k_{crest}^2$ PR: Power Ratio
[F28] Crest Factor	$k_{crest} = c(t_{MP}) / C_{RMS}$
[F29] NLOS Probability (by Decawave [23])	$prNLoS = function(\Delta t_{MP\_FP}, t_{MP})$ Formula provided by Decawave in [23] empirically based on laboratory and real-world tests.

Feature catalogue. We implement 29 features computed from the CIR, which are listed in Table I. The assigned identifiers (F1, F2, ...) already correspond to the ranked order of importance presented in §III-A and shown in Fig. 3. Most of the features are taken from literature on UWB [4], [5], [12], [14], [23]. We added a few features, which are modified versions from some features made relative to other related features by means of subtraction, F3: τ_{MED_L}, F8: ΔE_{CIR_FP}, F9: τ_{MED_FP}, F25: t_{rise_FP}, or division, F15: $ER_{FP:CIR}$, F17: $ER_{FP:MP}$, and adapted the Rician K Factor, F12 & F22: k_{RK}, from RSSI-based Wi-Fi sys-

tems [33], [34]. All five datasets contain CIR values, but some do not provide all the necessary per-measurement acquisition parameters (e.g., the noise power) to compute certain features (e.g., the SNR). Therefore, we use the CIR values to estimate the missing parameters.

Feature selection. The supervised feature selection (Fig. 2) is performed inside the HP tuning CV loop, in accordance with Section 7.10.2 of [35], i.e., only unsupervised screening steps can be performed outside of the CV loop. To select the best features, we divided this process into two pipelines: P3 and P4 (Fig. 2). In P3, we employ a Decision Tree (DT) model from the Python `scikit-learn` library to compute feature importance using the Gini impurity metric and create a ranking for feature selection [25], [26]. Only the best HP1 features according to this ranking are selected. High correlation between features can be disadvantageous for ML models such as SVM [14]. Therefore, in P4, features with an absolute pairwise correlation coefficient exceeding a threshold (HP2) are grouped together. One feature per group is selected based on a certain criteria (e.g., max. feature importance).

Reduction of the feature extraction time. Two methods for the reduction of the FE-Time by means of CIR-WL reduction and feature selection have been implemented: *(i)* through the HP tuning inside the ML Pipeline, and *(ii)* cross-datasets CIR-WL selection, by means of the KS statistic.

The *ML pipeline HP tuning* achieves the feature selection by choosing a single CIR-WL for feature extraction (HP0) along with the number of features (HP1). Although the FE-Time is not part of the HP tuning optimization objective, by selecting features and reducing the number of features and the CIR-WL, this method already provides lower FE-Times.

The second method, *cross-datasets CIR-WL selection*, allows the reduction of the FE-Time further, by choosing different CIR-WLs for each feature of the selected feature set. In a subsequent step to our feature selection process, the best per-feature CIR-WL is chosen based on the KS statistic. We present experimental results for this method in §III-B, performing feature selection without the ML Pipeline.

We use the KS statistic for feature selection as follows. Per feature, the KS statistic is a measure of the difference between the two Probability Density Functions (PDFs) of the feature for the two class labels (LoS and NLoS), and is calculated as the maximum absolute difference of their Cumulative Distribution Functions (CDFs). For balanced data, a single feature binary classifier theoretically achieves an accuracy of $50\% + KS \cdot 50\%$. Therefore, a higher KS statistic implies a higher predictive power of the feature. Then, to select a common set of features with differing CIR-WLs and achieve a very competitive accuracy on all datasets in our evaluation, our method proceeds as follows. For each feature, separately, the per-dataset KS statistics is maximized by fine-tuning the CIR-WL. We calculate, per CIR-WL, the mean of the feature's KS statistic across datasets. Then, the *optimal smallest* CIR-WL is chosen, which *approximately* achieves the *max* of the *mean* KS across datasets.

TABLE II
CHARACTERISTICS OF THE FIVE UTILIZED DATASETS.

Dataset	Characteristics	
ANGARANO [18], [37] Data size: 55k NLoS: 76,6%	- No device parameters - Short CIR-WL (157 Samples) - Only ranging error, with no ranging Ground Truth (GT)	! No Channel setting information - No CIR scale normalization + Variation of obstacles and rooms sizes ! Consideration of a plastic obstacle as LoS case
BREGAR #1 [16], [38] Data size: 42k NLoS: 50%	! Using Channel 2 ! Focus NLoS/LoS classes + Device parameters - No ranging GT	+ Many indoor locations - Randomized with respect to the indoor location
BREGAR #2 [17], [39] Data size: 15.5k NLoS: 68.5%	! Using all 5 channels ! Focus: ranging error mitigation + Device parameters + Ranging GT	+ Many indoor locations - Distance ranges per case: · LoS: 10 (1m to 4.5m); · NLoS: 25 (2m to 8m)
STOCKER CPS [5], [36] Data size: 36,7k NLoS: 33.6%	! Using Channel 5 + Device parameters + Ranging GT, with 230 distances per scenario and ranges of up to 12m	+ Many indoor locations + Three scenarios: LoS, NLoS, WLoS (32%)
(own) INDUSTRIAL Data size: 95,7k NLoS: 50,3%	! Using Channel 9 ! Industrial scenario with focus on MPCs + Device parameters	+ Ranging GT + 2 devices and 2 receivers + 96 distances per scenario
Legend	+ Good, - Negative,	! Important characteristics

ML classifiers. We explore multiple ML models, divided according to their usual memory size being greater or smaller than 2 kB, respectively, into *(i) large* models such as Support Vector Machine (SVM) with RBF kernel (RBFSVC), DTs, k-Nearest Neighbors (kNN), and single-Layer Perceptron Classifier (MLPC), and *(ii) small* ones such as Logistic Regression (LogReg), Linear SVM (LinSVC), linear model Ridge classifier (Ridge), and Gaussian Naïve Bayes (GaussianNB). This split separates small models, which are embeddable in small MCUs with only several kilobytes (kBs) of memory, from large models which often require MBs of memory.

Datasets. To analyze the best features and CIR-WLs, we exploit five UWB datasets, whose main characteristics are listed in Table II. The first four rows refer to publicly-available datasets. These datasets vary in size, proportion of NLoS/LoS measurements, scenarios, environments, availability of parameters important for the processing of the CIR, and implicit biases. The choice of UWB channel potentially affects the signal transmission, e.g., through obstacles. It is noteworthy that the channels differ between datasets, except for ANGARANO, which does not provide this information, but is stated to be using settings based on those from BREGAR #1.

When not specified, the CIR-WL in the dataset is 300 samples. We remark that the STOCKER CPS dataset [36] introduced a third scenario, named Weak-LoS (WLoS). However, similar to Stocker et al. [5], we decided to ignore the WLoS measurements and focus on NLoS/LoS classification to make our evaluation comparable across the five datasets.

The last row in Table II describes our own INDUSTRIAL dataset, which we acquired in an industrial warehouse setting using a robot. This dataset represents typical indoor industrial environments with metallic obstacles, NLoS conditions, and rich MPCs. Different from the 4 publicly-available datasets, which only use Decawave radios, INDUSTRIAL includes measurements with the MobileKnowledge platforms [40] featuring

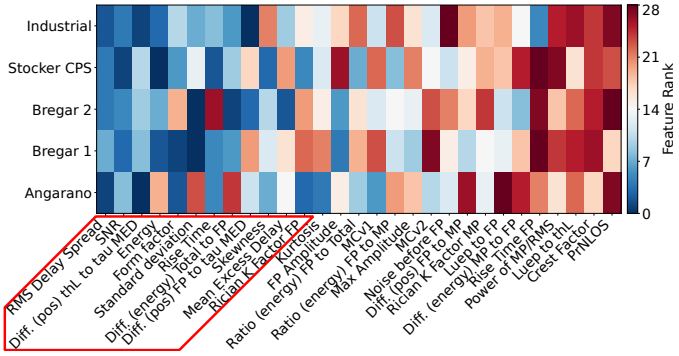

Fig. 3. Across-datasets feature importance ranking. The color coding shows the feature ranking per dataset, with features (x-axis) ordered by the overall rank computed across the five datasets (y-axis). The 12 best features are highlighted using a red frame.

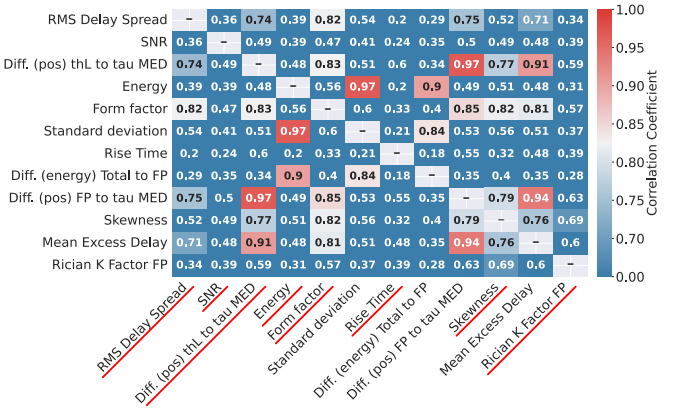

Fig. 4. Correlation matrix of the 12 best features by DT-based ranking. The color coding emphasizes highly correlated features. The 8 selected features (having a correlation threshold ≥ 0.85) are underlined in red.

NXP SR150 chipsets with two chip and three patch antennas. INDUSTRIAL provides 95k samples from 96 robot positions and with an NLoS/LoS ratio of ≈50%. The distance Ground Truth was acquired using the robot and an OptiTrack motion capture system [41].

III. EXPERIMENTAL RESULTS

We present the outcome of our methods for feature selection (§III-A) and reduction of the FE-Time (§III-B), as well as a performance evaluation across the five datasets of our small ML models and those trained using the ML pipeline (§III-C).

A. Feature Selection

We demonstrate the application of our feature selection method across the five datasets in Fig. 3 and 4. We apply the feature selection with the mean optimal values of HP1 and HP2 estimated by the Nested CV procedure and compute the feature importance ranking per dataset using a DT with depth 8 (see §II), which achieves good accuracy without overfitting. Interestingly, the best 12 (HP1) features are related to shape (i.e., the energy distribution over the CIR), energy, and timing characteristics of the CIR.

We observe that INDUSTRIAL, STOCKER CPS, and BREGAR #1 result in similar feature rankings, while ANGARANO and BREGAR #2 do not follow the same trend consistently (Fig. 3). We conjecture that is because in ANGARANO, there is *(i)* a lack of CIR normalization factors, *(ii)* a high percentage of NLoS cases, and *(iii)* different types of obstacles to create NLoS conditions, which results in the best ranked features to be timing- and shape- related, with the ranking of energy-related features being lower. In BREGAR #2, we achieve near perfect classification accuracy, 99.18% (kNN, §III-C), which hints the lack of certain corner cases present in other datasets.

Fig. 4 shows the correlation matrix of our top 12 features (Fs), which allows us to further reduce the number of features (#features) to 8, with an FE-Time of 45.8 ms, by using a correlation coefficient threshold of 0.85 (HP2). Features are selected from left to right, according to their average importance ranking across-datasets (Fig. 3), and rejecting features that present correlations ≥ 0.85 with the already selected set of features. Furthermore, using a 10-fold evaluation, we have estimated the accuracy per dataset of ML-models with increasing #features (Fig. 5), following our ranking and applying the correlation coefficient threshold. Competitive results across datasets can be achieved with only 3 Fs (over 84% accuracy and upwards, except in Angarano), and somewhat better results with 7 to 9 Fs (around 0.1-4% better accuracy in comparison to with 3 Fs, depending on the dataset and ML model), and only minor increases in accuracy with more Fs (<1%). Considering the pairwise-correlations (Fig. 4), we select our overall best 3 Fs with RMS-DS (F1), SNR (F2) and E_{CIR} (F4), which yields a FE-Time of 6.21 ms. τ_{MED_L} (F3) is not included, since it is correlated with RMS-DS.

Comparison to SoA. We compare our findings with the features selected by existing works, when provided in the literature, their relative ranking (we use as notation {F1, F2, F3, ...}, where F1 is the highest-ranked feature). We denote the distance estimated by Two-Way Ranging (TWR) as feature F100, as this is used by several related works [5], [12], [14] – yet, we refrain from using it because distance estimates are not available in all datasets nor localization systems using TDoA, i.e., this feature can hardly be used in practical settings.

The genetic algorithm by Zeng et al. [12] provides very different feature selections and rankings compared to the set of 12 top Fs we have identified, namely: {F100, F4, F18, F7, F6, F24}, {F4, F18, F7, F17}, and {F4, F18, F7, F6, F24, F17}. Interestingly, the three best features that we have identified (i.e., F1, F2, F3, see Figs. 3 and 4) have not been considered in their feature selection sets. Unfortunately, due to the unavailability of their dataset, we are unable to estimate the relative ranking of F1, F2, and F3 in their settings.

Bregar et al. [17] use F1, F4, F10, F11, F13 and F15·F4, which is a combination by means of multiplication of our features F15 and F4. Stocker et al. [5] use F8 and {F100, F13, F7, F11, F4, F1, F18, F20, F8}. Ferreira et al. [14] use F2, F4, F7+F25, F8, F14, F15 and F100. Notably, these three works did not include some of our best features, i.e., the SNR (F2), τ_{MED_L} (F3), k_F (F5), and σ_{CIR} (F6). Moreover, these works make all use of τ_{MED} (F11), and we show that F3 – which is a slight variation of F11 that is also strongly related to τ_{MED_L} – performs better.

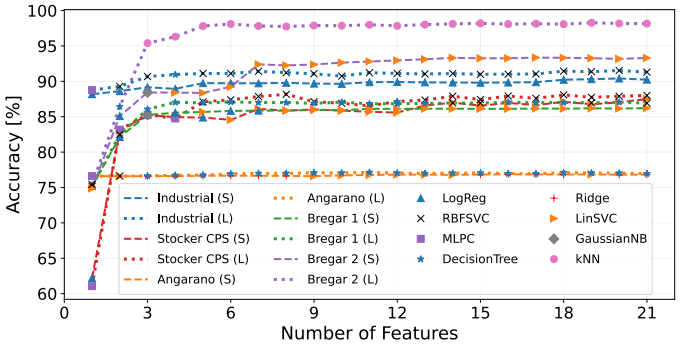

Fig. 5. Effect of the per-dataset feature selection, with 2 curves of the same color per dataset (S: Small, L: Large models), showing mean accuracy vs. number of features and best classifier (markers), 10-fold evaluation.

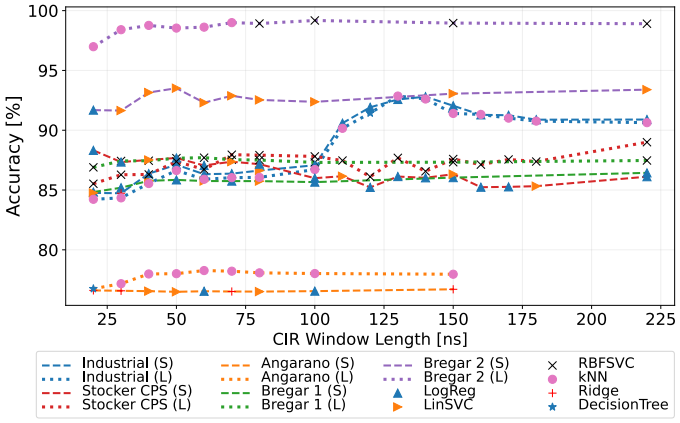

Fig. 6. Effect of CIR-WL, with 2 curves of 1 color per dataset (S: Small, L: Large models), showing mean accuracy vs. CIR-WL (feature extraction) and best classifiers (markers) from the Nested CV assessment.

This analysis highlights how existing works use largely different feature sets, and emphasizes the importance of our work, which is the first shedding light on which features perform best across *several* datasets. Moreover, when considering only a small numbers of features, the feature importance order we have derived yields better accuracy than the one proposed by the SoA (e.g., [5] consider F1 and F4 of lower importance in STOCKER CPS, which results in lower performance).

B. Feature Extraction Time Reduction

Feature selection provides the most relevant features sets to achieve competitive classification accuracy, while reducing the #features to extract and therefore the FE-Time. From the Nested CV assessment, we obtain the average accuracy per dataset of the ML-models using (mean values across folds) 5.8 to 12.4 features (see §III-C), all extracted by processing only a certain CIR-WL (Fig. 6). For all datasets except INDUSTRIAL, the accuracy does not depend strongly on the CIR-WL, and 50 ns yields competitive results.

In INDUSTRIAL, the classifiers perform better using features in the 120–150 ns CIR-WL range. Upon inspection of the affected samples, these features deliver new information for samples rich in MPCs. Our results indicate that using shorter CIR-WLs can impact both the computational efficiency and the accuracy. To showcase this effect, we estimate the efficacy

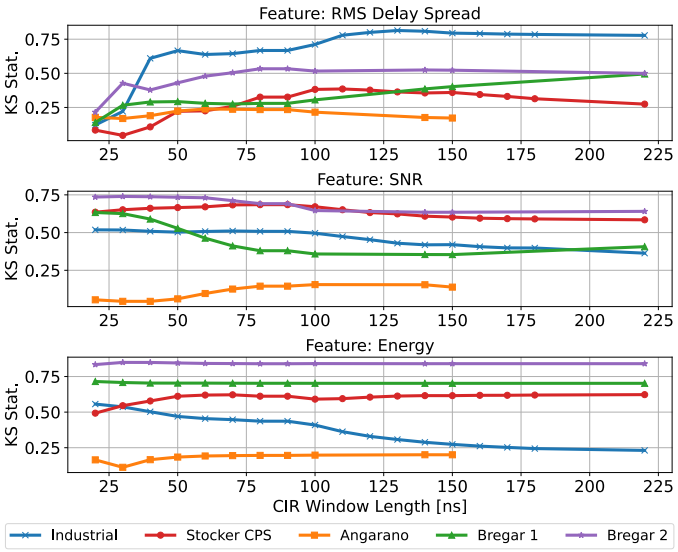

Fig. 7. Per-dataset efficacy for NLoS classification of the 3 best features, calculated using the KS statistic vs. the processed CIR-WL for each dataset.

of the features for the NLoS classification by means of the Kolmogorov-Smirnov (KS) statistic against the processed CIR-WL (Fig. 7) for our overall three best features: F1, F2 and F4. A single feature classifier would theoretically achieve an accuracy of $50\% + KS \cdot 50\%$ for NLoS/LoS balanced data. The best classification efficacy is often obtained for long CIR-WLs, but, as shown in Fig. 7, there are exceptions where shorter CIR-WLs work better (e.g., in the 120–150 ns range for INDUSTRIAL).

By applying the cross-datasets CIR-WL selection (discussed in §II), using all datasets except ANGARANO due to its shorter CIR-WL (Table II), to our best 8/3 Fs sets, we obtain the "CIR-WL-Opt" overall best 8/3 Fs sets. Following this procedure, we select the 3 Fs set: RMS-DS, SNR and energy, with "near-optimal" CIR-WLs of 140 ns, 60 ns and 40 ns respectively, which we denote F1(140), F2(60) and F4(40). We select the 8 Fs set with optimal CIR-WLs: F1(130), F2(30), F3(130), F4(30), F5(130), F7(120), F10(130) and F12(130). From the common CIR-WL (220 ns) to these varying CIR-WLs, we reduce the FE-Times for the 8 Fs and 3 Fs sets from 45.8 and 6.21 ms to 27.7 and 3.61 ms.

Comparison to SoA. Our "CIR-WL-Opt" 3 Fs set (FE-Time: 3.61 ms) achieves a reduction in the FE-Time of up to 95% (see Table IV) when compared to the SoA [5], [17].

C. NLoS Classification Results

In this section, we discuss our NLoS classification results and compare our achieved accuracies with the SoA. We present the performance characteristics of our ML models in Table III. The accuracy is used as the main performance metric for the NLoS/LoS classification models. Each main row of Table III corresponds to a different performance evaluation, described in the first column, using different sets of features and/or CIR-WLs. The FE-Time for the corresponding feature set is provided in the first column, except for the ML Pipeline, where this and other performance parameters of the resulting

TABLE III

MAIN PERFORMANCE METRICS ATTAINED ACROSS FIVE DIFFERENT DATASETS BY OUR BEST NLOS CLASSIFICATION ML MODELS[b].

	Dataset	ANGARANO [c]		BREGAR #1		BREGAR #2		STOCKER CPS		INDUSTRIAL	
		Best-L	Best-S	Best-L	Best-S	Best-L	Best-S	Best-L	Best-S	Best-L	Best-S
All 29 Fs CIR-WL: 220 ns[c] FE-T: 90 ms	ML-Model	DT	LinSVC	RBFSVC	LogReg	kNN	LinSVC	MLPC	LinSVC	RBFSVC	LinSVC
	Accuracy	77.41%	**76.70%**	87.10%	**86.52%**	98.73%	**93.73%**	88.95%	87.99%	91.87%	90.82%
	Mem-Size	28.52 kB	1.01 kB	2.77 MB	1.08 kB	3.21 MB	1.01 kB	9.12 kB	1.01 kB	3.93 MB	1.01 kB
	MP-Time	389 ns	412 ns	703 us	447 ns	203 us	1.15 us	1.15 us	924 ns	1.21 ms	336 ns
ML Pipeline[a] Nested CV 5-fold model assessment	ML-Model	kNN	LogReg	RBFSVC	LogReg	RBFSVC	LinSVC	RBFSVC	LogReg	kNN	LogReg
	Accuracy	**78.27%**	**76.70%**	**87.70%**	86.44%	**99.18%**	93.51%	**89.00%**	**88.32%**	92.86%	**92.80%**
	Mem-Size	15.53 MB	0.89 kB	0.89 MB	0.9 kB	51.57 kB	0.88 kB	0.25 MB	0.92 kB	25.44 MB	0.92 kB
	MP-Time	1.57 ms	308 ns	1.2 ms	1.51 us	45.3 us	748 ns	127 us	920 ns	379 us	882 ns
	FE-Time	3.29 ms	6.64 ms	3.42 ms	39.3 ms	5.8 ms	2.74 ms	22.3 ms	1.42 ms	5.16 ms	8.33 ms
	CIR-WL	60 ns	150 ns	60 ns	220 ns	100 ns	50 ns	220 ns	20 ns	130 ns	140 ns
	# Features	10.6	5.8	9.8	9	11.6	12.4	8.8	11.3	6.4	11.6
Overall best 8 Fs CIR-WL: 220 ns[c] FE-T: 45.8 ms	ML-Model	kNN	LogReg	RBFSVC	LogReg	kNN	LogReg	RBFSVC	LogReg	RBFSVC	LogReg
	Accuracy	77.74%	76.58%	87.37%	86.10%	98.67%	90.71%	88.49%	85.96%	91.39%	89.65%
	Mem-Size	7.41 MB	0.91 kB	0.89 MB	0.89 kB	2.18 MB	0.89 kB	0.4 MB	0.89 kB	1.31 MB	0.89 kB
	MP-Time	109 us	256 ns	598 us	407 ns	45 us	913 ns	259 us	642 ns	789 us	183 ns
Overall best 8 Fs CIR-WL-Opt FE-T: 27.7 ms	ML-Model	kNN	LogReg	RBFSVC	LogReg	kNN	LinSVC	RBFSVC	LinSVC	RBFSVC	LogReg
	Accuracy	77.28%	76.57%	87.07%	84.65%	98.14%	89.18%	87.73%	84.30%	**93.56%**	91.48%
	Mem-Size	8.01 MB	0.91 kB	0.91 MB	0.87 kB	2.15 MB	0.81 kB	0.34 MB	0.84 kB	0.95 MB	0.89 kB
	MP-Time	163 us	414 ns	605 us	455 ns	74.2 us	1.76 us	211 us	640 ns	638 us	306 ns
Overall best 3 Fs CIR-WL: 140 ns FE-T: 4.17 ms	ML-Model	RBFSVC	Ridge	DT	LogReg	kNN	LinSVC	MLPC	LogReg	RBFSVC	LogReg
	Accuracy	76.61%	76.61%	86.21%	84.80%	95.89%	88.23%	84.67%	84.49%	92.69%	91.32%
	Mem-Size	1.03 MB	0.88 kB	29.19 kB	0.84 kB	0.89 MB	0.78 kB	6.6 kB	0.87 kB	0.63 MB	0.87 kB
	MP-Time	1.21 ms	210 ns	273 ns	306 ns	22.7 us	598 ns	504 ns	459 ns	719 us	128 ns
Overall best 3 Fs CIR-WL-Opt FE-T: 3.61 ms	ML-Model	RBFSVC	Ridge	RBFSVC	LogReg	kNN	LinSVC	MLPC	LogReg	RBFSVC	LogReg
	Accuracy	76.61%	76.57%	86.13%	84.53%	95.79%	87.99%	83.99%	84.00%	92.32%	90.96%
	Mem-Size	1.02 MB	0.88 kB	0.49 MB	0.84 kB	0.89 MB	0.78 kB	6.7 kB	0.87 kB	0.66 MB	0.87 kB
	MP-Time	1.21 ms	209 ns	589 us	747 ns	24.57 us	719 ns	462 ns	378 ns	818 us	314 ns

[a]ML Pipeline: of the parameters shown, {ML-model, and CIR-WL} are fixed by the HP tuning, whereas {Accuracy, Mem-size, MP-Time, FE-Time, #Fs} are averages over the 5 model assessment folds. [b]Per column, the best accuracy is indicated in bold, and the models with the fastest FE-Times are underlined. In terms of the MP-Time and the ML-Models that we tested, the reader can assume that all small models and the DT (depending on the tree depth) have similar MP-Times in a small MCU, whereas for the large models a measurement of the MP-Time in the MCU is required. [c]The longest CIR-WL contained in ANGARANO is 157 ns, which results in FE-Times of 64.5 ms (for all 29 Fs, row 1) and 30.6 ms (for overall best 8 Fs, row 3).

TABLE IV

COMPARISON OF THE ACHIEVED ML-MODEL ACCURACY AND EXTRACTION TIME USING PROPOSED AND SOA FEATURE SETS.

Dataset	State of the Art (SoA)[d]					StrGrSplit	Proposed Approach[e]				Diff. SoA to Proposed	
	Paper	ML-Model	#Fs	FE-T[f]	Accuracy[g]	Accuracy[h]	ML-Model	#Features	FE-T[f]	Accuracy	ΔAccuracy	ΔFE-T
BREGAR #1 [38]	[17]	CNN	-	-	87.40%	-	RBFSVC	(Nested CV) 9	3.42	**87.70%**	0.3%	-
		LinSVC	6	76.60	82.50%	**85.87%**	LinSVC	(CIR-WL-Opt) 3	3.61	84.33%	-1.54%	-95.29%
		DT	6	76.60	-	**86.43%**	DT	(WL 140 Best) 3	4.17	86.21%	-0.22%	-94.56%
STOCKER CPS [36]	[5]	RBFSVC	3	32.50	63.00%	76.10%	LogReg	(CIR-WL-Opt) 3	3.61	**84.00%**	7.90%	-88.89%
		RBFSVC	7	41.60	84.00%	86.66%	LogReg	(Nested CV) 11	1.42	**88.32%**	1.66%	-96.59%
		RBFSVC	8	41.67	90.00%	87.46%	LogReg	(Nested CV) 11	1.42	**88.32%**	0.86%	-96.59%
		RBFSVC	9	42.30	92.00%	**88.36%**	LogReg	(Nested CV) 11	1.42	88.32%	-0.04%	-96.64%
		LogReg	3	32.50	-	72.79%	LogReg	(CIR-WL-Opt) 3	3.61	**84.00%**	11.21%	-88.89%
		LogReg	9	42.30	-	82.58%	LogReg	(CIR-WL-Opt) 3	3.61	**84.00%**	1.42%	-91.46%
		DT	9	42.30	-	**84.09%**	DT	(CIR-WL-Opt) 3	3.61	82.45%	-1.64%	-91.46%

[d]FE-Time estimated using the full CIR-WL of 210 ns and our benchmark on the 80 MHz ESP8266 MCU. [e]We do not include the distance estimated by Two-Way Ranging (TWR) as a feature (F100). [f]FE-T is provided in ms. [g]Stocker et al. [5] used the F1 score as accuracy metric. [h]A re-evaluation via 10-fold CV, using our data preparation (§II) is provided for the feature sets of the SoA [5], [17]: (i) to provide other researchers with a solid basis for the comparison of NLoS classification results, and (ii) to better analyze the performance of our feature selection method. For (ii), we compare different feature sets, keeping all other factors that affect classifier accuracy the same. When re-evaluating the feature sets of Stocker et al. [5], following their methodology, we hyper-tuned the RBFSVC model parameters.

ML models are calculated using average values over the 5 model assessment folds (for more details, see footnote '*a*' of Table III). The rest of the columns, starting from the third column, provide per-dataset performance results, where each column is splitted in two, corresponding to the best *large* and *small* models (respectively Best-L and Best-S).

Table III contains the following performance parameters, as specified in its second column: the ML classifier type (ML-Model), the average accuracy in the test sets across the CV folds, the memory size, the prediction time of the model (MP-Time), the FE-Time (or FE-T), the CIR-WL, and the number of features (mean across model assessment folds). The FE-Time per ML-Model is calculated by adding up the FE-Times of its features. Per column, the best accuracy is indicated in bold, and the models with the fastest FE-Times are underlined. All performance evaluations have been done via 10-fold CV, except for the Nested CV 5-fold model assessment (§II).

Per CIR-WL feature extraction time. We measure the per-feature extraction time per CIR-WL using an 80MHz ESP8266 MCU as a reference system, which allows us to estimate corresponding FE-Times in other embedded systems. Fig. 8 shows that, for most features, the computational complexity increases with the CIR-WL. P_{NBFP} (F20), which is not shown in the figure, is the only exception, with a constant computation time of $\approx 70\,\mu$s independently of the CIR-WL. We analyze next how to further reduce the FE-Time by shortening the processed CIR-WL for feature extraction. We measure the prediction times per ML-Model on a computing

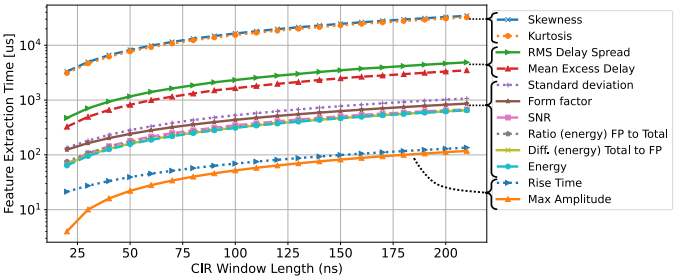

Fig. 8. Impact of the CIR-WL reduction on the feature extraction time (FE-time). The FE-time scale is shown in logarithmic form. The legend is arranged from slowest to fastest, top to bottom. This data highlights the high FE-Times required for some features and the need for making a judicious selection of features and their corresponding CIR-WLs.

cluster, and derive the ML classifier model size as the size of the model saved locally on a 64-bit system.

ML pipeline. The Nested CV procedure allows us to automatically select the best performing combinations of #features and CIR-WL per-dataset and per-ML-method. This is supported by the fact that the models derived with the ML pipeline often achieve the best performance in Table III. In many cases, thanks to strong reductions of the CIR-WL, the derived models require lower FE-Times than those proposed by existing work.

Results across five datasets. Table III shows that the performance of the models derived by the ML pipeline and of those using only the overall best 3/8 Fs are rather similar throughout all studied datasets. Moreover, our 3/8 Fs models using "CIR-WL-Opt" achieve an accuracy comparable to their full CIR-WL counterparts and to the ML pipeline models at much faster FE-Times, which shows the effectiveness of our solution. In the INDUSTRIAL dataset, our small models benefit from the use of a CIR-WL of 140 ns and achieve an accuracy above 90%. In fact, as shown in Fig. 6, this CIR-WL allows to deal with the presence of MPCs in some corner cases. Unsurprisingly, also the models derived using Nested CV tend to select features in the 120–150 ns CIR-WL range.

Comparison to SoA. By comparing the performance of the models shown in Table III with those of existing work, we can further appreciate the effectiveness of our proposed approach. For example, in the STOCKER CPS dataset, Stocker et al. [5] achieve 63% F1 score when using 3 Fs only (see Table IV, footnote '*g*'), whereas we achieve an accuracy of 84% with 3 Fs, and of 88.32% with the fastest small model. Stocker et al. [5] achieve 92% F1 score when using 9 Fs, but those include F100, which – as discussed in §III-A – is omitted in our models due to its limited practicality, and they do not use the *stratified group k-fold split* method. For BREGAR #1, we outperform [17] (which proposed a CNN achieving 87.4% accuracy and a 6 Fs LinSVC model achieving 82.5% accuracy) with a faster RBFSVC model with 10 Fs and 60 ns CIR-WL (87.70% accuracy). Compared to the 6 Fs LinSVC model in [17], we could also achieve a higher accuracy (86.21%) with a very fast 3 Fs DT. In Table IV, we explicitly compare our results to the SoA, based on our feature sets with reduced FE-Times discussed in §III-B. The SoA accuracy column shows results reported in the papers [5], [17], whereas the *StrGrSplit accuracy* column shows our re-evaluations of the feature sets from the SoA (see Table IV, footnote '*h*') The re-evaluations with matching ML-Models allow us to compare the performance of the feature sets of the SoA against our feature sets, by keeping all other factors that affect classifier accuracy the same. Therefore, they enable us to analyze the performance of our feature selection. Our re-evaluations show that our feature sets and feature selection methods achieve accuracy almost on par with the SoA, while achieving 90% faster FE-Times. Specifically, our methods result in Fs sets with 3–11 Fs that offer competitive accuracy compared to the SoA (from 1.64% worse up to 11.21% better), even with small ML-Models, while reducing FE-Times by 89–97%.

Results with small ML models. Compared to the Model Assessment (Table III), we achieve competitive results across datasets, and therefore retain great classification accuracy, using the "CIR-WL-Opt" common sets of 3 and 8 features. This demonstrates the robustness of both feature sets to classify competitively corner cases on all datasets. These tiny models, or NLoS classifiers, feature a size of ≈1 kB and very low computation times (≈3.6 and 27.7 ms on the 80 MHz ESP8266 MCU), and are hence competitive candidates for implementations on resource-constrained UWB devices (such as the Qorvo `MDEK1001`) that require a high update rate.

Key takeaways. We want to highlight the following takeaways from this work: *(i)* for ML models implementing manually-crafted features, performing feature selection while taking FE-Times into consideration is crucial to achieve competitive runtimes; *(ii)* our feature selection method implementing the tree-based Gini impurity metric for feature importance and the subsequent reduction of the selected features by means of the pairwise feature correlations results in competitive accuracy in all datasets (§III-C); *(iii)* while features in our catalogue (Table I) are related to shape, energy, and timing characteristics of the CIR, for NLoS classification shape and energy related features have made the top of our feature inclusion ordering for most datasets based on our feature selection method; *(iv)* the best features are very dataset-dependent and our best 3 features in average are not the best 3 features for any dataset alone (as shown in Fig. 3); *(v)* owing to our comprehensive benchmarking across five datasets, our "CIR-WL-Opt" sets of 3 and 8 features should provide competitive results in new datasets; *(vi)* there might be corner cases which benefit from using features with specific CIR-WLs, such as ≈140 ns for INDUSTRIAL (Fig. 6), which can be studied extracting the KS statistics on the new dataset; *(vii)* although the FE-Time is not part of the HP tuning optimization objective, the ML Pipeline already provides lower FE-Times (Table IV); and *(viii)* our results in Table III show that large models (except MLPC) require 30 to 25000 times more memory than small models and *(ix)* for MP-Time, kNN and RBFSVC require 15 to 12000 times longer as DTs, MLPCs, and smaller models.

Recommendations. Owing to our comprehensive benchmarking across five datasets, our results can be applied to new datasets as follows: *(i)* our "CIR-WL-Opt" sets of 3 and 8

features should provide competitive results; *(ii)* our extensive feature catalogue with its ranking or feature inclusion ordering (Table II and Fig. 3) enables competitive feature selection by performing a parameter sweep on the number of features during the ML model HP Tuning (similarly to Fig. 5); *(iii)* a measurement of the FE-Time in the target hardware allows to skip high FE-Time features (e.g, F10, F13, see Fig. 8); and *(iv)* the calculation of the features' pairwise correlations (e.g., Fig. 4) and KS statistics (e.g., Fig. 7) on the new dataset enable reduced FE-Times while usually maintaining accuracy by, respectively, reducing the selected number of features and by choosing optimal values for the features' CIR-WLs.

IV. Conclusions

Our unbiased model assessments on four publicly available datasets, obtained through Nested CV using the ML pipeline, achieve very competitive results in comparison to the SoA and provide other researchers with a solid basis for the comparison of NLoS classification results. Our tiny 3-feature NLoS classifiers outperform classifiers with similar #features from the SoA and showcase very low computation and FE-Times of ≈3.6 ms in a resource-constrained MCU with a memory size of ≈1 kB, while achieving competitive accuracies between 76.61% and 90.96% depending on the dataset and achieving a reduction of the runtime of more than 90% compared to the SoA. These tiny models are promising candidates for the implementation of NLoS classification in low-power UWB tags.

As future work, we will extend this study by also covering NLoS error mitigation in addition to classification. We further plan to improve our feature selection method by incorporating the feature extraction time as a part of the HP tuning within the ML Pipeline. Furthermore, we intend to merge datasets to obtain models that generalize better and narrow the gap between datasets and the real-world implementation.

Acknowledgments

This work has been supported by the FFG-COMET-K1 Center "Pro²Future" (Products and Production Systems of the Future), Contract No. 881844, within the ENHANCE-UWB project.

References

[1] A. Ledergerber *et al.*, "A Robot Self-Localization System using One-Way UWB Communication," in *Proc. of the IROS Conference*, 2015.

[2] T. Brovko *et al.*, "Complex Kalman Filter Algorithm for Smartphone-based Indoor UWB/INS Navigation Systems," in *Proc. of the IEEE USBEREIT Symposium*, 2021.

[3] EETimes, "VW and NXP Show First Car Using UWB To Combat Relay Theft," 2019. https://www.eetimes.com/volkswagen-and-nxp-show-first-car-using-uwb-to-combat-relay-theft/ – Last access: 2023-08-01.

[4] S. Maranò *et al.*, "NLOS Identification and Mitigation for Localization based on UWB Experimental Data," *IEEE JSAC*, vol. 28, no. 7, 2010.

[5] M. Stocker *et al.*, "Performance of Support Vector Regression in Correcting UWB Ranging Measurements under LOS/NLOS Conditions," in *Proc. of the 4th CPS-IoTBench Workshop*, 2021.

[6] J. Borras *et al.*, "Decision Theoretic Framework for NLOS Identification," in *Proc. of the 48th IEEE VTC Conference*, 1998.

[7] V. Minh Le *et al.*, "Human Occlusion in UWB Ranging: What Can the Radio Do for You?," in *Proc. of the 18th MSN Conference*, 2022.

[8] I. Guvenc *et al.*, "NLOS Identification and Mitigation for UWB Localization Systems," in *Proc. of the IEEE WCNC Conference*, 2007.

[9] M. Gallacher *et al.*, "InSight: Enabling NLOS Classification and Error Correction on Resource-Constrained Ultra-Wideband Devices," in *Proc. of the 20th EWSN Conference*, 2023.

[10] S. Venkatesh *et al.*, "Non-line-of-sight Identification in Ultra-wideband Systems based on Received Signal Statistics," *IET MAP*, 2007.

[11] K. Yu *et al.*, "Statistical NLOS Identification based on AOA, TOA, and Signal Strength," *IEEE Trans. on Vehicular Technology*, vol. 58, 2008.

[12] Z. Zeng *et al.*, "UWB NLOS Identification with Feature Combination Selection based on Genetic Algorithm," in *IEEE ICCE Conf.*, 2019.

[13] T. Van Nguyen *et al.*, "Machine Learning for Wideband Localization," *IEEE Journal on Selected Areas in Communications*, 2015.

[14] A. Ferreira *et al.*, "Feature Selection for Real-Time NLOS Identification and Mitigation for Body-mounted UWB Transceivers," *IEEE Trans. on Instrumentation and Measurement*, vol. 70, 2021.

[15] Decawave, "DW1000 User Manual, Version 2.11," 2017.

[16] K. Bregar *et al.*, "NLOS Channel Detection with Multilayer Perceptron in Low-Rate Personal Area Networks for Indoor Localization Accuracy Improvement," in *Proc. of the 8th IPSSC Conference*, 2016.

[17] K. Bregar and M. Mohorčič, "Improving Indoor Localization Using Convolutional Neural Networks on Computationally Restricted Devices," *IEEE Access*, 2018.

[18] S. Angarano *et al.*, "Robust Ultra-wideband Range Error Mitigation with Deep Learning at the Edge," *Elsevier EAAI*, 2021.

[19] J. Fontaine *et al.*, "Edge Inference for UWB Ranging Error Correction Using Autoencoders," *IEEE Access*, 2020.

[20] H. Brunner *et al.*, "Understanding and mitigating the impact of wi-fi 6e interference on ultra-wideband communications and ranging," in *ACM/IEEE IPSN'2022*, 2022.

[21] B. Jones *et al.*, "Tiny but Mighty: Embedded Machine Learning for Indoor Wireless Localization," in *Proc. of the 20th CCNC Conf.*, 2023.

[22] V. Savic *et al.*, "Measurement Analysis and Channel Modeling for TOA-based Ranging in Tunnels," *IEEE TWC*, 2014.

[23] Decawave, "Application Note APS006 (Part 2): Non Line of Sight Operation and Optimizations to Improve Performance in DW1000 based Systems, Version 1.5," 2014.

[24] V. Kumar and S. Minz, "Feature Selection: A Literature Review," *Smart Computing Review*, vol. 4, no. 3, 2014.

[25] M. R. Al Iqbal *et al.*, "Knowledge Based Decision Tree Construction with Feature Importance Domain Knowledge," in *IEEE ICECE*, 2012.

[26] K. Topouzelis and A. Psyllos, "Oil Spill Feature Selection and Classification using Decision Tree Forest on SAR Image Data," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 68, 2012.

[27] M. Stocker *et al.*, "Applying NLOS Classification and Error Correction Techniques to UWB Systems: Lessons Learned and Recommendations," in *Proc. of the 6th CPS-IoTBench Workshop*, 2023.

[28] C. Kar and A. Mohanty, "Application of KS Test in Ball Bearing Fault Diagnosis," *Journal of Sound and Vibration*, vol. 269, 2004.

[29] M. Lewis *et al.*, "Nested Cross-Validation," 2023. [Online] https://cran.r-project.org/web/packages/nestedcv/vignettes/nestedcv.html.

[30] A. Vabalas *et al.*, "Machine Learning Algorithm Validation with a Limited Sample Size," *PLOS ONE*, vol. 14, no. 11, 2019.

[31] D. Krstajic *et al.*, "Cross-Validation Pitfalls when Selecting and Assessing Regression and Classification Models," *Springer Journal of Cheminformatics*, vol. 6, 2014.

[32] S. Varma and R. Simon, "Bias in Error Estimation when using Cross-Validation for Model Selection," *BMC Bioinformatics*, 2006.

[33] Z. Xiao *et al.*, "Non-line-of-sight Identification and Mitigation using Received Signal Strength," *IEEE Trans. on Wireless Comm.*, 2014.

[34] S. Cotton and W. Scanlon, "A Statistical Analysis of Indoor Multipath Fading for a Narrowband Wireless Body Area Network," in *Proc. of the 17th IEEE PIMRC Symposium*, 2006.

[35] T. Hastie *et al.*, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2^{nd} ed., 2009.

[36] M. Stocker *et al.*, "Stocker CPS UWB NLOS Dataset," 2021. [Online] http://iti.tugraz.at/uwb-nlos – Last access: 2023-08-01.

[37] S. Angarano *et al.*, "DeepUWB Dataset," Nov. 2020. [Online] https://doi.org/10.5281/zenodo.6611037 – Last access: 2023-08-01.

[38] K. Bregar *et al.*, "Bregar #1 nlos/los dataset," 2016. https://github.com/ewine-project/UWB-LOS-NLOS-Data-Set – Last access: 2023-08.

[39] K. Bregar *et al.*, "Bregar #2 Loc. dataset," 2018. https://github.com/ewine-project/UWB-localization/tree/master/data/ – Last Acc.: 2023-08.

[40] MobileKnowledge, "MK UWB Kit SR150/SR040," 2023. [Online] https://www.themobileknowledge.com/product/mk-uwb-kit-sr150-sr040/ – Last access: 2023-08-01.

[41] OptiTrack, "OptiTrack – Motion Capture (MoCap) Systems," 2023. [Online] https://optitrack.com – Last access: 2023-08-01.