Poster: Towards NLOS Ranging Error Detection and Mitigation using Machine Learning on Embedded Ultra-Wideband Devices

Markus Gallacher, Michael Stocker, Carlo Alberto Boano, and Kay Römer
Institute of Technical Informatics, Graz University of Technology, Austria
{markus.gallacher, michael.stocker, cboano, roemer}@tugraz.at

Abstract

We study the ranging error classification and mitigation capabilities of machine learning models used in ultra-wideband systems. This is relevant, as distance estimates in non-line-of-sight (NLOS) conditions can be off by several meters, which may severely compromise the performance of applications that require location awareness. Our ultimate goal is to optimize the size of a convolutional neural network (CNN) used for classifying and mitigating ranging errors such that it can run on constrained embedded devices without affecting its performance. To this end, we present an optimized CNN implementation that, in contrast to resource-hungry machine learning models requiring hundreds of kB of memory, can classify and mitigate NLOS conditions with 12 kB of RAM and 75 kB of ROM.

1 Motivation and Goals

Ultra-wideband (UWB) technology uses a high bandwidth ($\geq 500MHz$), which results in a high time resolution and allows for centimeter-level ranging accuracy [4]. UWB has become the preferred distance estimation technology for indoor localization applications such as asset tracking, assisted living, and secure keyless vehicle access. Despite the good performance of UWB in standard line-of-sight (LOS) conditions, where ranging accuracy is typically within $10-20cm$, some challenges remain in non-line-of-sight (NLOS) conditions. In fact, obstacles that attenuate or block the first (i.e., direct) path can significantly increase the ranging error by up to a few meters, as the time of flight (ToF) of the signal is prolonged due to slower propagation in different materials, or signal reflections. This undesirable effect is typically tackled in two ways: either via NLOS *classification* or *mitigation*. Classification is used to detect NLOS conditions and to simply ignore these measurements. This approach works well when there are multiple nodes to range with and erroneous measurements taken in NLOS conditions can be dropped. The other option is to directly mitigate the induced ranging error by correcting it. This approach is more suitable for infrastructures with a limited number of devices, where no alternative node can be used. Related works [2] use statistical approaches to detect or mitigate NLOS conditions, but often require knowledge of the surroundings (e.g., floor map), or work with a small amount of features (e.g., variance of range estimates and channel bandwidth). For this reason, machine learning (ML) approaches are gaining popularity to find more general solutions using more complex features.

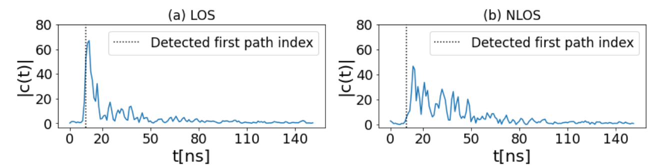

Figure 1. Exemplary CIR for LOS and NLOS conditions. The first path is delayed and attenuated due to obstacles.

The channel impulse response (CIR) provided by the UWB nodes can be analyzed to receive insights about the quality of the ranging estimate. Specifically, the CIR shows the amplitudes over time of the first path of the signal pulse and all its reflections (i.e., multi-paths), as shown in Fig. 1. With the help of ML models, the information extracted from the CIR can be used to classify or mitigate NLOS conditions. CNNs are the neural network of choice when dealing with spatially-independent data (e.g., objects in images or amplitude positions in audio), and have grown in popularity to classify or mitigate NLOS conditions based on information extracted from the CIR [1]. CNNs can automatically extract spatial features from the data while reducing the overall complexity by alternating convolutional layers (feature extractions) and average or maximum pooling layers (average or maximum of a pool of nodes).

Bringing CNNs to constrained embedded platforms. Unfortunately, ML models, especially CNNs, tend to be resource-hungry and computationally expensive. They typically do not run on constrained embedded devices such as the popular UWB node MDEK1001[1], which embeds 32 kB of RAM and 256 kB of flash memory. We study how to adapt an exemplary CNN, called *REMNet* [5], to fit on an MDEK1001, while still achieving adequate performance.

[1]https://www.qorvo.com/products/p/MDEK1001

2 Optimized *REMNet* Implementations

REMNet [5] is a regression model that predicts the ranging error and uses the latter to correct the ranging estimate.

Baseline. We create a *REMNet* baseline implementation by combining the classification approach presented by Bregar et al. [3] with the *REMNet* regression solution proposed by Angarano et al. [5], and by changing the *REMNet* output layer to also perform classification. We use both *REMNet* models to perform classification and regression and refer to it as `C&R`. We also adapt the *REMNet* to be a *multi-output* network that can perform classification and regression simultaneously; we will refer to it as `MO`. This multi-output *REMNet* has two output neurons: one returning the predicted error, and the other classifying the measurement as LOS or NLOS. Therefore, only one model is needed instead of two separate ones. To practically build lightweight models, we use the TensorFlow Light library for microcontrollers, which provides the basic set of functionalities to implement neural networks. We adjust some layers of the *REMNet* to be compliant with the library and provide CIRs with 152 samples as input, starting 10 samples before the detected first path, as shown in Fig. 1. We will refer to these models as the *baseline*.

Full integer quantization. To reduce the memory requirements and computation time, we use full integer quantization, i.e., we only use integer values for all the model's internal calculations. We will refer to these models as *quantized*.

Model optimization. Based on the *quantized* models, we perform a grid search with different hyperparameters to evaluate the performance on even smaller models (which we will call *optimized*). We consider as hyperparameters: the CIR length provided as an input (L), the number of residual reduction modules that extract features and reduce the dimensionality of the data (N), and the number of filters applied in each layer to extract features (F).

3 Preliminary Evaluation

We evaluate our optimized *REMNet* implementations and compare their performance with that of the baseline `C&R` and `MO` models. We use the dataset by Stocker et al. [4], containing measurements from 749 settings with 50 traces each.

Dataset. We divide the settings into two groups: the first one consists of LOS traces recorded without any obstacles between devices. The second group consists of NLOS traces with either small ranging errors of a couple of decimeters (mostly from smaller obstacles) or large ranging errors of a couple of meters (mostly from walls). In our evaluation, we select a uniformly-distributed set of LOS and NLOS traces to avoid bias. Our results are obtained by performing *k-fold* cross validation with three folds and by repeating this procedure 3 times with a newly-shuffled dataset.

Performance metrics. We quantify the performance of the optimized models with three metrics. The $F1$ *score* in the range 0–1 defines how well the classification models classify new data (with 1 being the best value). The R^2 *score* defines how well the regression models' error prediction matches the true ranging error (the perfect score would have a value of one, but it can be arbitrarily worse). The *model size* considers the classification and regression TFLite micro-models combined, and roughly corresponds to the RAM footprint.

Table 1. Performance of baseline and optimized models.

	Type	L	N	F1 score	R^2 score	Model size
MO	*Baseline*	152	3	0.77	0.05 ($\pm$0.23)	39 *kB*
	Quantized	152	3	0.76	0.04 ($\pm$0.29)	27 *kB*
	Optimized	152	1	0.75	0.13 ($\pm$0.17)	14 *kB*
	Optimized	32	1	0.76	0.25 ($\pm$0.09)	12 *kB*
C&R	*Baseline*	152	3	0.76	0.26 ($\pm$0.12)	76 *kB*
	Quantized	152	3	0.76	0.28 ($\pm$0.17)	52 *kB*
	Optimized	152	1	0.72	0.29 ($\pm$0.08)	24 *kB*
	Optimized	32	1	0.75	0.13 ($\pm$0.36)	23 *kB*

Results. Table 1 illustrates our results: the *quantized* model alone already reduces the model size by 30%, while maintaining similar R^2- and F1-scores. The performed grid search shows that the hyperparameter F should not be reduced, as the R^2-score decreases on average by 40% and 90% for $F = 8$ and $F = 4$, respectively, and is therefore kept constant. Reducing the hyperparameter N further reduces the *quantized* model size by about 50% without affecting the F1-score, while actually improving the R^2-score. We believe that higher N values cause overfitting during regression, which leads to a high standard deviation and a worse R^2-score. The reduced CIR length hyperparameter has a positive effect on the `MO` model, by greatly reducing the standard deviation and increasing the mean of the R^2-score. We also quantify the ROM usage of the proposed models, and observe a ROM footprint of 237, 164, and 75 kB for `MO` baseline, quantized, and optimized models (N=1, L=32), respectively. The ROM footprint for the `C&R` baseline, quantized, and optimized models (N=1, L=32) is 452, 317, and 139 kB, respectively.

4 Conclusion and Outlook

We can reduce the model size of classification and regression CNNs by up to 65% thanks to full integer quantization and other hyperparameter optimizations, while still achieving a similar prediction performance. The models fit on embedded devices with less than 32 *kB* of RAM. In future work, we plan to run our solution directly on the `MDEK1001` UWB platform, and to explore and compare other ML models.

Acknowledgements. This work was performed within the SPiDR project and was partially supported by the TU Graz LEAD project "Dependable IoT in Adverse Environments".

5 References

[1] A. Nessa, B. Adhikari, F. Hussain and X. N. Fernando. A Survey of Machine Learning for Indoor Positioning. *IEEE Access*, 8:214945–214965, 2020.

[2] J. Khodjaev, Y. Park and A. S. Malik. Survey of NLOS Identification and Error Mitigation Problems in UWB-based Positioning Algorithms for Dense Environments. *Annales des Télécommunications*, 65:301–311, 2010.

[3] K. Bregar and M. Mohorčič. Improving Indoor Localization Using Convolutional Neural Networks on Computationally Restricted Devices. *IEEE Access*, 6:17429–17441, 2018.

[4] M. Stocker, M. Gallacher, C. A. Boano and K. Römer. Performance of Support Vector Regression in Correcting UWB Ranging Measurements under LOS/NLOS Conditions. In *Proc. of the CPS-IoTBench '21 Workshop*, 2021.

[5] S. Angarano, V. Mazzia, F. Salvetti, G. Fantin and M. Chiaberge. Robust Ultra-Wideband Range Error Mitigation with Deep Learning at the Edge. *Engineering Applications of Artificial Intelligence*, 102:104278, 2021.